

AD-A052 055

CHARLES STARK DRAPER LAB INC CAMBRIDGE MA
A MICROPROCESSOR CONTROLLED MOTOR CONTROLLER FEASIBILITY STUDY. (U)
MAR 78 A L GULOVSEN

F/6 9/5

N00014-77-C-0352

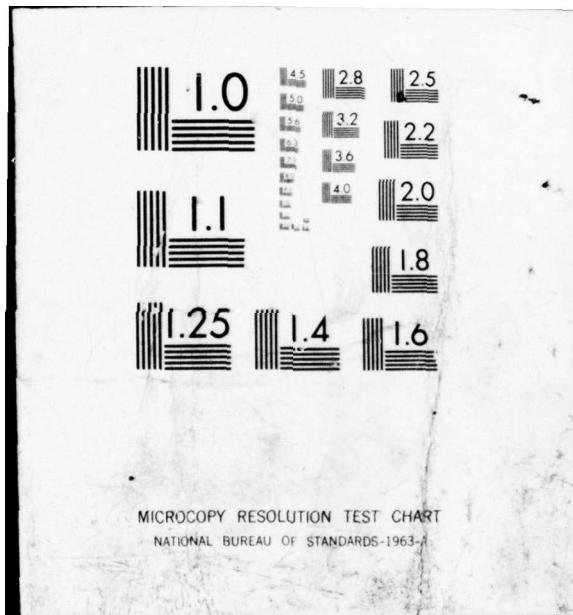
NL

UNCLASSIFIED

R-1147

1 OF 2
AD
A052 055





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-

12

AD A 052055

R-1147

A MICROPROCESSOR CONTROLLED
MOTOR CONTROLLER
FEASIBILITY STUDY

by

Arthur L. Gulovsen

January 1978

AD No.
DDC FILE COPY



The Charles Stark Draper Laboratory, Inc.

Cambridge, Massachusetts 02139

Approved for public release; distribution unlimited.

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle)		5. TYPE OF REPORT & PERIOD COVERED Final Report Dec 9 75 - Jan 9 78
A MICROPROCESSOR CONTROLLED MOTOR CONTROLLER FEASIBILITY STUDY		6. PERFORMING ORG. REPORT NUMBER R-1147
7. AUTHOR(s)	8. CONTRACT OR GRANT NUMBER(s) N00014-77-C-0352 rev	
9. PERFORMING ORGANIZATION NAME AND ADDRESS The Charles Stark Draper Laboratory, Inc. Cambridge, MA		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS David W. Taylor Naval Ship R & D Center Annapolis, Maryland 21402		12. REPORT DATE Mar 9 78
		13. NUMBER OF PAGES 95
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
David W. Taylor Naval Ship R & D Center Annapolis, Maryland 21402		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Microprocessor Control Algorithm Micro-Computer Simulation		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) A microprocessor controlled motor controller feasibility study has been completed by the Charles Stark Draper Laboratory for the U. S. Navy. In addition to verification of the concept of a microprocessor as the controlling element of a motor control system, design goals included the hardware realization of a control algorithm which maximizes system performance by minimizing harmonic, transient and steady state current, and also maximizes		

408386

JB

20. Abstract (Continued)

system reliability by limiting the stresses and overloads to each component.

REVIEWED BY
SAC - LOS ANGELES
APR 19 1968
FBI - LOS ANGELES
SEARCHED INDEXED SERIALIZED FILED
FBI - LOS ANGELES
APR 19 1968
FBI - LOS ANGELES

REVIEWED BY
SAC - LOS ANGELES
APR 19 1968
FBI - LOS ANGELES
SEARCHED INDEXED SERIALIZED FILED
FBI - LOS ANGELES
APR 19 1968
FBI - LOS ANGELES

REVIEWED BY
SAC - LOS ANGELES
APR 19 1968
FBI - LOS ANGELES
SEARCHED INDEXED SERIALIZED FILED
FBI - LOS ANGELES
APR 19 1968
FBI - LOS ANGELES

TMN/MV/CLM/MSA

R-1147

A MICROPROCESSOR CONTROLLED
MOTOR
CONTROLLER FEASIBILITY STUDY

by

Arthur L. Gulovsen

January 1978

Approved:

F. Houston
F. Houston

The Charles Stark Draper Laboratory, Inc.
Cambridge, Massachusetts 02139

ACCESSION for	
NTIS	White Section <input checked="" type="checkbox"/>
DDC	Buff Section <input type="checkbox"/>
UNANNOUNCED	
JUSTIFICATION _____	
IV	
DISTRIBUTION/AVAILABILITY CODES	
ALL and	Civil
A	

ACKNOWLEDGEMENT

This report was prepared by The Charles Stark Draper Laboratory, Inc. under Contract N00014-77-C-0352 with the Naval Ship Research and Development Center of the U.S. Navy.

Publication of this report does not constitute approval by the U.S. Navy of the findings or conclusions contained herein. It is published for the exchange and stimulation of ideas.

TABLE OF CONTENTS

<u>Paragraph</u>		<u>Page</u>
1	SUMMARY-CONCLUSIONS.....	1
2	CONTROL ALGORITHM BASIC DESCRIPTION.....	2
2.1	Variable Frequency Control.....	2
2.2	Variable Pulse Per Cycle Control.....	2
2.3	Transient Speed Control.....	3
2.4	Slip Control.....	3
2.5	Output Frequency (Motor Speed Limit).....	4
3	BASIC SYSTEM DESCRIPTION.....	5
4	SOFTWARE DESCRIPTION.....	7
4.1	Overall Program Organization.....	7
4.1.1	Parameter Calculations.....	7
4.1.2	Real Time Generation of Phase, Frequency and Pulse Width.....	9
4.1.3	Data Definition.....	10
4.2	Subroutine Description.....	13
4.2.1	Initial Conditions.....	13
4.2.2	Input Subroutine.....	13
4.2.3	IFOLD.....	14
4.2.4	Sc Foldback.....	15
4.2.5	Set Condition.....	15
4.2.6	Fc Hysteresis.....	16
4.2.7	Adj FCREG, PC, PW.....	17
4.2.8	Slip Control.....	17
4.2.9	Adj PW for B+.....	18
4.2.10	Calc. PWCNT and PWREG.....	18

TABLE OF CONTENTS (Cont.)

<u>Paragraph</u>	<u>Page</u>
4.2.11 Calc. FCCNT and FCREG.....	18
4.2.12 Test Output.....	18
4.2.13 Transfer Information.....	19
4.2.14 Multiply Subroutine.....	19
4.2.15 Divide Subroutine.....	19
4.2.16 GENPH - Generate Phase Pointer and Phase.....	19
4.2.17 Output Information.....	20
 5 HARDWARE DESCRIPTION.....	21
5.1 TTY Interface, Card A1.....	21
5.2 Computer Card A2.....	22
5.3 Computer Card A3.....	22
5.4 Output Interface Card A5.....	23
5.5 SCR Driver Card.....	25
5.6 Interrupt Generation.....	25
5.7 Power Stages.....	25
 6 SYSTEM SIMULATION.....	26
6.1 General Description.....	26
6.2 Model Organization.....	26
6.3 FORTRAN Model Program.....	27
6.4 Simulation Results.....	27
6.5 Simulation Curves.....	28
 APPENDIX A	A-1
Model Program Block Diagram.....	A-2
Model Program Printout Sheets.....	A-3
Model Program Simulation Curves.....	A-21

LIST OF ILLUSTRATIONS

<u>Figure</u>		<u>Page</u>
1	System Block Diagram.....	6
2	Initial Conditions.....	8
3	Outer Loop Subroutine Flow.....	30
4	Input Information.....	31
5	IFOLD.....	32
6	SC FOLDBACK B+,IAVE,TEMP.....	33
7	Set Conditions.....	34
8	FC Hysterisis.....	35
9	Adj. FCREG, Determine PC & PW	36
10	SLIPC.....	37
11	Adjust PW for B+(100V to 128V Range).....	38
12	PWPT.....	39
13	FC Pointer SR.....	40
14	Output Test Data.....	41
15	Transfer Information.....	42
16	Divide Subroutine.....	43
17	Multiply Subroutine.....	44
18	Overview of Inner Loop.....	45
19	Inner Loop 1.....	46
20	Inner Loop 2.....	47
21	Inner Loop 3.....	48
22	Inner Loop 4.....	49
23	MODEL PROGRAM BLOCK DIAGRAM: SIMULATION PROGRAM FLOW..A-2	

SECTION 1

SUMMARY-CONCLUSIONS

1.1 Introduction

The Microprocessor controlled Motor Controller Feasibility study was conducted by The Charles Stark Draper Laboratory (CSDL) from December 1975 to January 1978. Design goals included the hardware realization of a control algorithm which maximizes system performance by minimizing harmonic, transient and steady state current, and also maximizes system reliability by limiting the stresses and overloads to each component. Verification of the concept of using a microprocessor as the controlling element of a motor control system was another goal.

1.2 Conclusions

Variable frequency control is a proven efficient method of controlling the speed of ac induction motors. Variable pulse per cycle control and transient speed control, as implemented in this microprocessor controlled motor controller system, provided significant improvement to induction motor speed control. The slip control and output frequency limit algorithm also improved motor speed system performance, to a lesser degree.

All design goals established for this feasibility study were attained.

A microprocessor controlled motor controller was designed, fabricated and debugged by CSDL and the microprocessor was programmed and debugged. The results of tests conducted with an ac induction motor verified improved motor performance and showed the concept of using a microprocessor for this type of application a definite success.

A FORTRAN model of the motor and controller was generated, and a simulation conducted on the CSDL Amdahl computer. The results gave further proof of the feasibility of the overall concept.

SECTION 2

CONTROL ALGORITHM BASIC DESCRIPTION

2.1 Variable Frequency Control

Variable frequency control has been proven as an efficient method of controlling the speed of ac induction motors. Frequency is easily varied over an 8 to 1 range and could be increased if desired. A constant volt-time product (modified by slip control routine) must be maintained at the motor input over the frequency range. This requirement causes some secondary problems and the control algorithm has been designed to alleviate them.

The best motor control performance results from incorporation of the following major features into the control algorithm:

- (1) Variable pulse per cycle control
- (2) Transient speed control
- (3) Slip control
- (4) Output frequency limit

The advantages of each feature are described below.

2.2 Variable Pulse Per Cycle Control

Pulse width modulation is used to maintain a constant volt-time product over the frequency range. With this type control the percent modulation varies directly with frequency while the harmonic content of the waveform is a function of the percent modulation. The harmonic content of the waveform at a particular frequency is a function of the number of pulses per cycle. By increasing the number of pulses per cycle in the lower frequency ranges, a reduction in the 5th and 7th harmonics may be achieved.

- (a) Frequency of 45 to 60 Hz 1 pulse/cycle full pulse width
- (b) Frequency of 30 to 45 Hz 2 pulse/cycle 1/2 pulse width
- (c) Frequency of 10 to 30 Hz 4 pulse/cycle 1/4 pulse width
- (d) Starting at 7.5 to 10 Hz 8 pulse/cycle 1/8 pulse width

Not only is the harmonic content reduced thereby reducing the heat dissipation of the motor, but motor cogging, a problem in the past, is also eliminated by applying more pulses of smaller pulse width during lower speed conditions.

2.3 Transient Speed Control

Starting and reversing motor currents can be 5 to 6 times full load running currents if maximum frequency is instantaneously applied or plug reversing of the motor is allowed. Starting and reversing motor currents may be reduced to 2.5 times full load running current if the frequency change applied to the motor is limited to a specified increment above or below the present motor speed. This approach essentially slews the motor to the desired operating speed. The torque created by the motor is greater than that of the brute force approach, therefore, the motor can be started and reversed faster while drawing less current.

It must be emphasized that direct plug reversing of a motor should be definitely avoided when designing a motor controller system.

2.4 Slip Control

Slip is determined by comparing the frequency applied to the motor with actual motor speed measured by a tachometer. Slip is usually controlled by varying the applied voltage, but in this case slip is controlled by varying the pulse width.

The torque-speed characteristic of a motor varies as the square of the applied voltage at a particular frequency. If the slip is too small (motor running close to synchronize speed) the applied voltage (pulse width) is decreased thereby lowering the torque-speed motor curve causing the motor to run slower. If the slip is too large, the pulse width is increased, increasing the torque-speed curve causing the motor to run faster.

The prime advantage of this type control is a reduction in motor current made possible by an improvement in power factor. With a motor running at ideal slip its power factor is about 0.85. Power factor decreases fairly rapidly at low slip approaching zero at synchronous speed, therefore, by controlling the pulse width applied to the motor its torque-speed curve is effected causing the motor to run close to its ideal slip. The result is a higher power factor and a lower motor current.

2.5 Output Frequency (Motor Speed) Limit

Motor temperature and average motor current are monitored. If exceeding a specified limit, the applied frequency is reduced. Alarms and motor shut downs would occur at still higher temperatures and currents. The applied frequency is similarly limited if the primary voltage source (a battery in this application) drops below a given value. Pulse width is reduced at higher battery voltage resulting in a constant voltage-time product.

SECTION 3

BASIC SYSTEM DESCRIPTION

Figure 1 is a block diagram of the implementation of a control algorithm. The following is a description of the major elements.

The micro-computer's (microprocessor + memory) program is entered and altered by the operator via the teletype unit. Speed command, motor speed, motor current, motor temperature, and battery voltage are signal conditioned and then input to the micro-computer via the I/O circuitry. The micro-computer calculates frequency, number of pulses per cycle and pulse width and performs the real time generation of phase, frequency and pulse width control which are output via the I/O circuitry. The output interface electronics performs the additional timing and interlocks necessary for SCR commutation and control. The gate drive circuits provide the ground isolation and gate current drive required to turn the SCR's on. The SCR power stages deliver the 3 phase power to the motor.

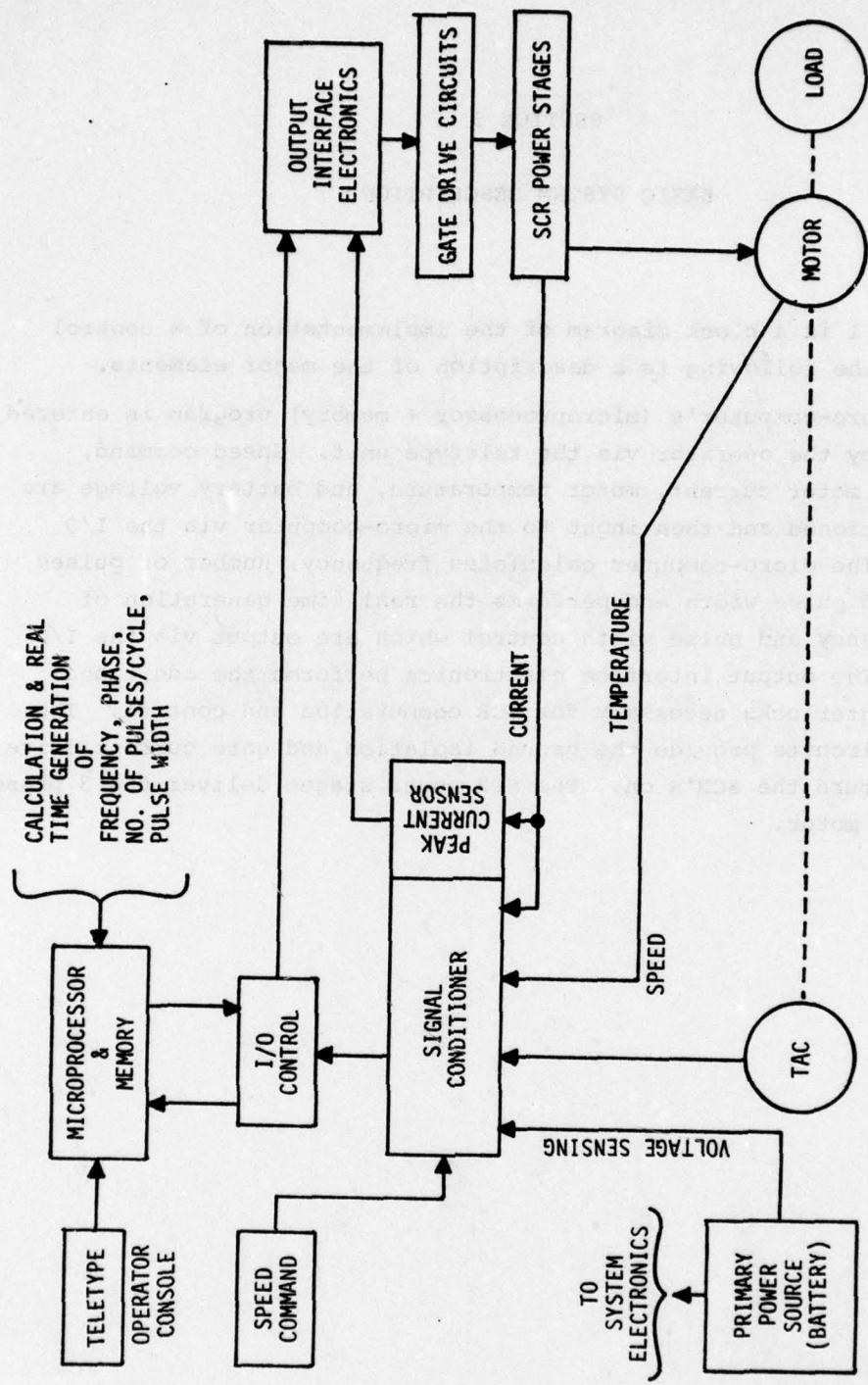


Figure 1. System block diagram.

SECTION 4

SOFTWARE DESCRIPTION

4.1 Overall Program Organization

The program is organized into an inner and outer loop configuration in which three main tasks are performed.

- (1) Calculation of output frequency (FC), pulse width (PW); and the number of pulses per cycle (PC), based on input conditions, is performed in part of the outer-loop.
- (2) The inner-loop under interrupt control performs the real time generation of phase frequency and pulse width.
- (3) An additional calculation is performed in the outer loop which converts FC and PW to a set of parameters which is required for the function of the inner-loop.

The mnemonics for the microprocessor (BSR, JSR, RTS, etc.) are defined in the literature furnished with the processor and are not included in this report. The program flow diagram (Figure 2), should be referenced when reading the descriptions.

4.1.1 Parameter Calculations

- (1) Input the present conditions
 - (a) Operator speed command (Sc)
 - (b) Motor speed (TAC)
 - (c) Motor current and Temperature (IAVE & Temp)
 - (d) Battery voltage (BPL)
- (2) Calculate: based on input information
 - (a) Output frequency command (Fc)
 - (b) Number of pulse per cycle (# P/C)
 - (c) Width of each pulse (PW)

INITIAL CONDITIONS

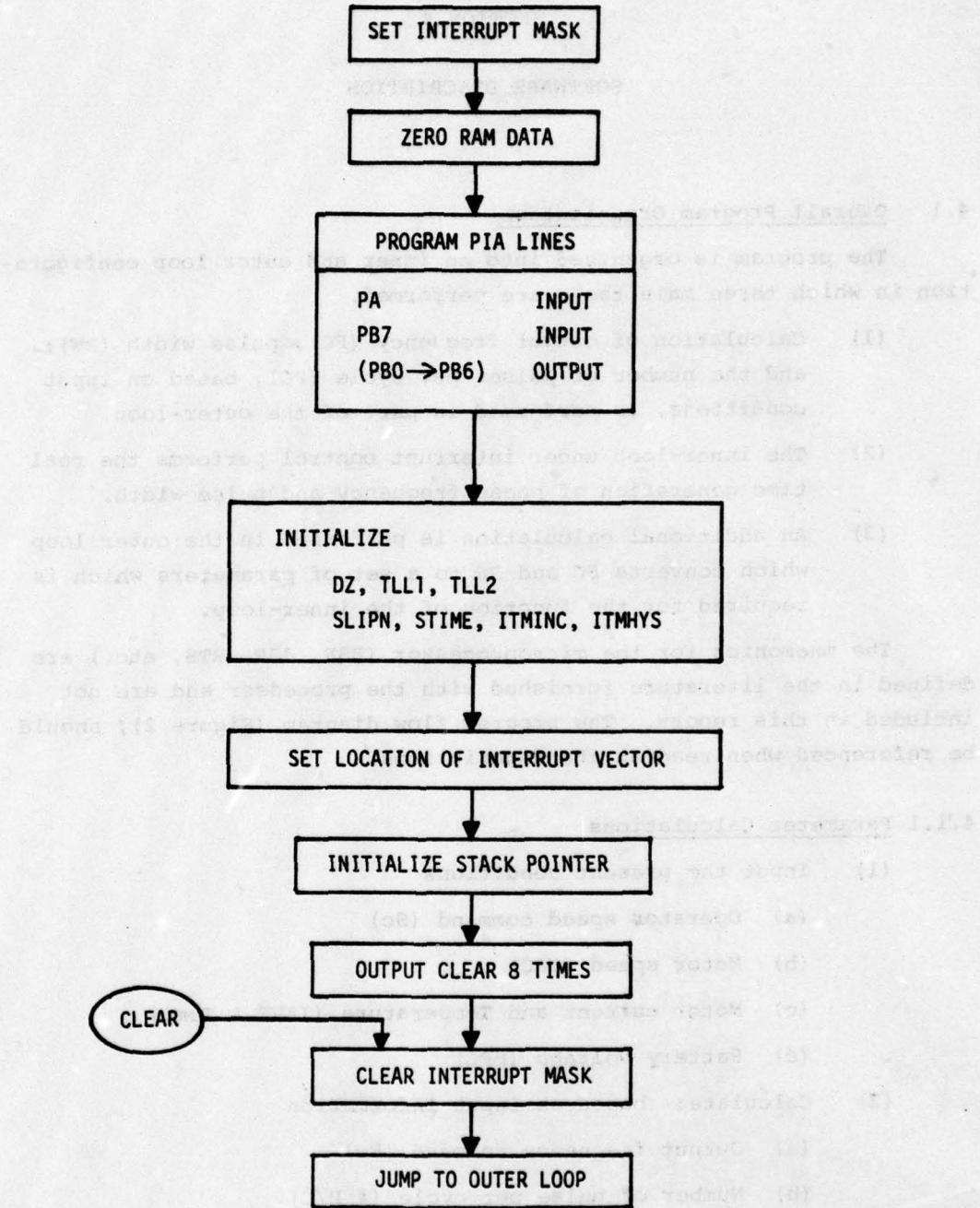


Figure 2. Initial conditions.

(3)* Calculate: based on pulse width.

(a) Pulse width counter (PWCNT).

The number of interrupts until the pulse width generator is evoked.

$$\text{PWCNT} \approx \text{PW/interrupt time.}$$

(b) Pulse width register (PWREG).

The time remaining to be counted down by the inner-loop pulse width generator.

$$\text{PWREG} = \text{PW} - \text{PWCNT} \times \text{interrupt time.}$$

(4)* Calculate: based on frequency and pulses per cycle.

(a) Frequency counter (FCCNT).

The number of interrupts until the frequency generator is evoked.

$$\text{Period} = \frac{1}{\text{FC} \times \text{PC}}$$

$$\text{FCCNT} = \frac{\text{Period}}{\text{Interrupt time}}$$

(b) Frequency register (FCREG).

Represents the time remaining to be counted down by the inner-loop frequency generator.

$$\text{FCREG} = (\text{Period} - \text{FCCNT} \times \text{interrupt time}) \times \text{FC.}$$

*Note: There is a delay time (INDEL) which is subtracted from PW and Period in the above calculations. The delay is associated with the microprocessor interrupt and program execution times.

4.1.2 Real Time Generation of Phase, Frequency, and Pulse Width

(1) Frequency generation (FCGEN).

A timer which generates the period of the multiple pulse output frequency ($\text{FC} \times \text{PC}$).

FCCNT is decremental after each interrupt. When the FCCNT equals zero the FC generator is evoked which counts down FCREG. When FCREG equals zero the processor outputs ϕA , ϕB , ϕC and PULSE commands.

(2) Pulse width generation (PWGEN).

PWCNT is decremental after each interrupt. When PWCNT equals zero the PW timer is evoked which counts down PWREG. When PWREG equals zero the processor outputs ϕA , ϕB , ϕC and PULSE commands.

- (3) Three phase generation: (Based on # P/C and sign of Fc).
(GENPH)
Determines if there is to be a phase change during the next pulse width cycle and the direction of phase rotation. The phase generation routine is executed following the frequency rate generator routine.
- (4) Short pulse width generations (SHTPW).
Same as pulse width generation except it immediately follows the frequency generation executed when pulse width time is less than one interrupt plus delay.
- (5) Inner loop program flow.
 - (a) PWGEN routine
 - (b) FCGEN routine
 - (c) SHTPW routine
 - (d) GENPH routine
 - (e) RTI (return from interrupt)

4.1.3 Data Definition

Input Data

- (1) SC Speed Command magnitude input (1 bit = 1/4 Hz)
- (2) SCSGN Speed Command sign input (CW/CCW)
- (3) TAC Tachometer feedback magnitude input (1 bit = 1/4 Hz)
- (4) TACSGN Tachometer feedback sign input (CW/CCW)
- (5) BPL Battery voltage magnitude (1 bit = 1/2 V)
- (6) BPLSGN Battery voltage sign
- (7) IAVE Average motor phase current magnitude
- (8) IAVSGN Average motor phase current sign
- (9) TEMP Motor temperature magnitude
- (10) TEMSGN Motor temperature sign

Output Data

- (1) Phase plus Pulse Command 4 bits of information - (QA, OB, QC, PULSE). Output occurs at the start and stop of pulse width (PW). Output rate is a function of frequency command (Fc) and number of pulses per cycle (PC)

(2) Test Internal program data (selected by operator) is outputted to a D/A.
Used for program evaluation

Internal Data

- (1) PW Pulse width (1 bit = 10 microseconds)
- (2) PWCONT Number of interrupts prior to evoking PW timer
- (3) PWREG Pulse width time remaining to be counted down by PW timer
- (4) FC Frequency
- (5) DLTF Used in FC hysteresis calculation
- (6) FCCNT Number of interrupts prior to evoking FC generator
- (7) FCREG Used with FC in the FC generator to real time frequency output
- (8) FCREGH FCREGH used during FCCNT and FCREG calculations. FCREGH and FCREG are loaded with \$104 = 260 2600 microseconds is the period of the maximum output frequency
- (9) FCINTR Used during FCCNT and FCREG calculations
FCINTR = interrupt time x FC
- (10) FCDEL Used during FCCNT and FCREG calculations
FCDEL = frequency delay time x FC
- (11) FCREM Used during FCCNT and FCREG calculations
FCREM is the lower 8 bit remained when FCINTR is calculated.
- (12) Phase A three bit code which represents phase information
- (13) PHPONT Points to one of six memory locations which store phase information
- (14) PC Number of pulses per cycle - really the number of pulses per phase used in phase generated routine for incrementing PHPONT.
- (15) R Remainder - used with PC in phase generation routine
- (16) SCTAC Speed command minus TAC
- (17) DZ Speed command dead zone - has hysteresis
- (18) TLL1 TAC dead zone (motor starting) - has hysteresis

(19) TLL2 TAC dead zone (motor running) - has hysteresis.

(20) SLIPN Used in slip control routine to increment the effects of (SCTAC/IDEAL SLIP) on pulse width.

(21) STIME Used as delay in incrementing SLIPN in slip control routine.

(22) ISCALE Used in frequency foldback due to current (IFOLD) routine. A time delayed increment of FC due to high current.

(23) ITMHYS Used in IFOLD - equals either 1 or 2 seconds (hysteresis). High currents must be present for 2 seconds before foldback of frequency may occur. Enables normal current transients during motor slowing up or down.

(24) ITIME Used in IFOLD. ITIME equals the length of time. IAVE is greater than the current limit. ITIME must be greater than ITMHYS before IFOLD routine may be evoked.

(25) ITMINC Used in IFOLD. ITMINC causes a 50-millisecond time delay in the incrementing of ISCALE.

(26) MLTCAN Used in multiply routine.

(27) DENOM Used in divide routine.

(28) Zero False data for programming convenience.

(29) Zero 1 False data for programming convenience.

(30) data n →data being calculated in the outer loop
data X →data transferred from outer to a transfer state
data I →data being used by inner loop

Outer loop	Xfr loop	Inner loop
PWCNT	PWCNTX	PWCNTI
PWREG	PWREGX	PWREGI
FCCNT	FCCNTX	FCCNTI
FCREG	FCREGX	FCREGI
FC	FCX	FCI
SCSGN	SCSGNX	SCSGNI
PC	PCS	PCI

PHASE I

* PHASE is generated within the inner loop.

4.2 Subroutine Description

The following paragraphs describe the subroutines of the overall program. The referenced figures are flow diagrams for the subroutine.

4.2.1 Initial Conditions Routine

a. All memory data is initialized to zero except:

Dz = 8 Hz
TLL1 = 3 Hz
TLL2 = 5 Hz
SLIPN = MAX
STIME = 10 ms
ITMINC = 50 mx
ITMHYS = 2 sec

b. PIA is programmed:

PIAPA = Input
PIAPB7 = Input (Sign)
PIAPB(0+6) = Output
PIAPB(4,5,6) for decoding.
PIAPB(0,1,2,3) for data.

PB6	0	0	0	0	1	1	1	1
PB5	0	0	1	1	0	0	1	1
PB4	0	1	0	1	0	1	0	1
	Sc	TAC	B+	IAVE	Temp	Q+PW	Test LO Order	Test HI Order
INPUT							OUTPUT	

c. Output clear command 8 times

Clear is defined as ϕA , ϕB , & ϕC all low. Therefore no frequency is applied. The motor is at a stand still, but the commutation capacitors are charged.

4.2.2 Input Subroutine

a. Speed command, tachometer, B+, Average current, and Temperature are input to the processor and stored in memory.

- b. The sign of all inputs are tested and the magnitude complemented if negative.
- c. The index register is used to load the input data into the correct location in memory.

4.2.3 IFOLD

The function of this routine is to limit long term motor current below a prespecified limit by reducing the speed command. Motor accelerating and decelerating currents exceed the current limit but do not cause a SC foldback unless they exist for a period longer than 2 seconds.

When the current limit is exceeded ITIME is incremented and compared to ITMHYS. ITMHYS is initialized to 2 seconds. If ITIME is less than ITMHYS the foldback of SC is bypassed. If ITIME exceeds ITMHYS, ITMHYS is set to 1 second.

Once ILIM and ITMHYS have been exceeded, SC foldback due to IAVE begins. The method of SC foldback is to compare SC to the ratio $\left(\frac{IAVE - ILIM}{IMAX - ILIM}\right)$ and if the input SC is greater, set SC to $\left(\frac{IAVE - ILIM}{IMAX - ILIM}\right)$. It is necessary to implement SC foldback in an incremental and time delayed fashion, otherwise large current transients due to rapid SC changes would result. The delay is implemented by preloading ITMINC to 25 and decrementing each pass through the outer loop. Assuming 2 milliseconds per pass, this results in a 50 milliseconds delay.

The incrementing of SC is implemented through the calculation of ISCALE.

If ISCALE is greater than $\left(\frac{IAVE - ILIM}{IMAX - ILIM}\right)$

then $ISCALE = ISCALE - \left[\left(\frac{IAVE - ILIM}{IMAX - ILIM} \right) \right] - ISCALE / 8$.

If ISCALE is less than $\left(\frac{IAVE - ILIM}{IMAX - ILIM}\right)$

then $ISCALE = 7/8 ISCALE$.

With time $ISCALE \approx \frac{IAVE - ILIM}{IMAX - ILIM}$.

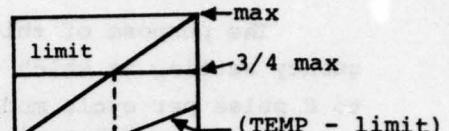
When current falls below ILIM and ITIME is less than ITMHYS, ISCALE is reduced as follows $ISCALE = ISCALE/2$.

4.2.4 Sc Foldback

- a. B+ and Temperature are tested for exceeding a prespecified limit. If so, speed command is limited.
- b. A clear command is generated if an undervoltage condition exists.
- c. Example of TEMP limit:

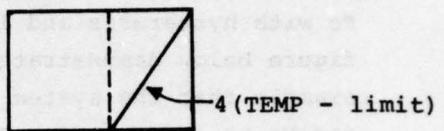
1) Subtract limit for TEMP

1a) Skip to end if less than limit

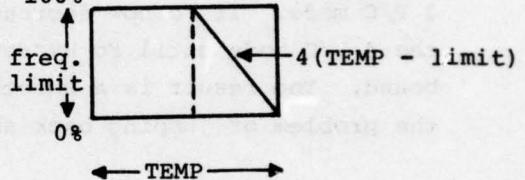


2) Scale (TEMP - limit)

Multiply by 4



3) Complement 4(TEMP - limit) 100%



4.2.5 Set Condition

Frequency command F_c is generated as a result of series of tests which determine if the system is in a steady state, transient (starting, stopping or reversing) or zero rate condition.

If steady state condition $F_c = S_c$

If transient starting condition $F_c = T_{AC} + DLTUP$

If transient stopping condition $F_c = T_{AC} - DLTDW$

If transient reversing condition 1st $F_c = T_{AC} - DLTDW$
(sign = TAC sign)

2nd $F_c = T_{AC} + DLTUP$
(sign = S_c sign)

If zero rate condition Clear command issued

$DLTUP = 7.5 \text{ Hz}$

$DLTDW = 3 \text{ Hz}$

Hysteresis is incorporated to prevent chattering between the various conditions

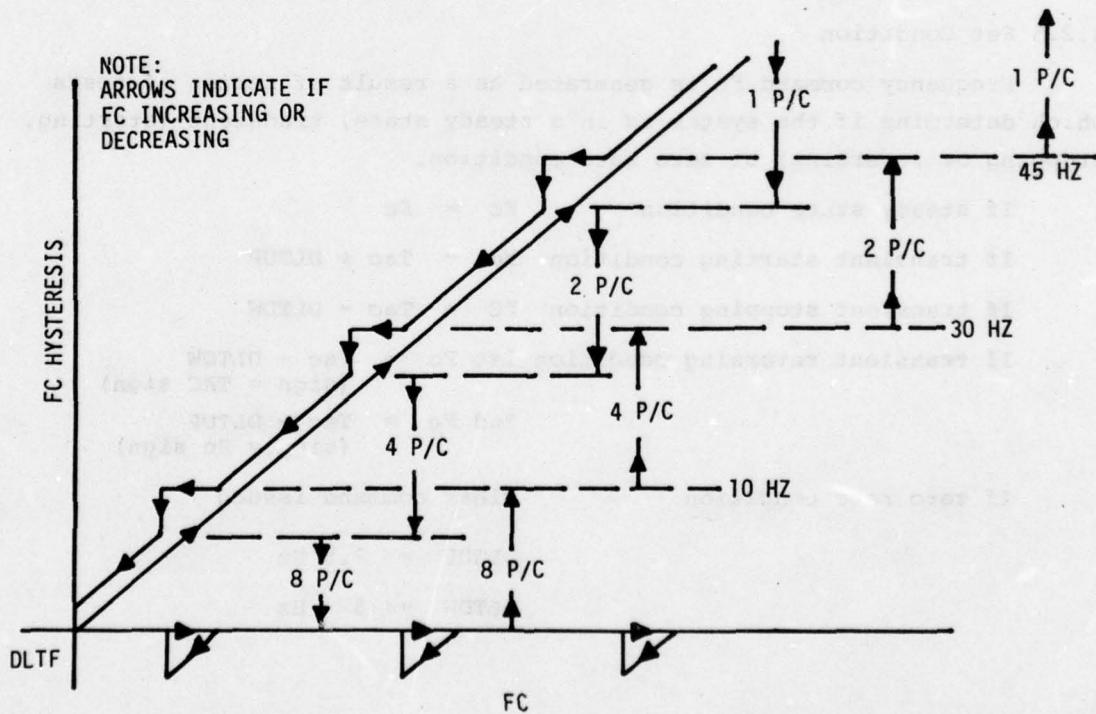
Sc dead zone ($DZ = 7.5$ and 8 Hz)

Tac low level ($TLL_1 = 2.5$ and 3 Hz)

 ($TLL_2 = 5$ and 5.5 Hz)

4.2.6 Fc Hysteresis

The purpose of this routine is to insert hysteresis in the frequency setting at which the system switches from 1 to 2, 2 to 4, and 4 to 8 pulse per cycle mode (P/C). This is accomplished by inserting hysteresis in the frequency command (Fc), determining P/C based on Fc with hysteresis and later removing the hysteresis from Fc. The figure below demonstrates the operation of this scheme. Consider for example that the system is operating in the 4 pulse/cycle (4 P/C) mode and Fc is increasing. The system will remain in the 4 P/C mode until Fc Hysteresis passes 30 Hz at which time the system switches to the 2 P/C mode. If Fc now decreases, the system will not switch back to the 4 P/C mode until Fc Hysteresis passes through the calculated lower bound. The result is a smooth transition of Fc from 0 to 60 Hz, while the problem of jumping back and forth between P/C is eliminated.



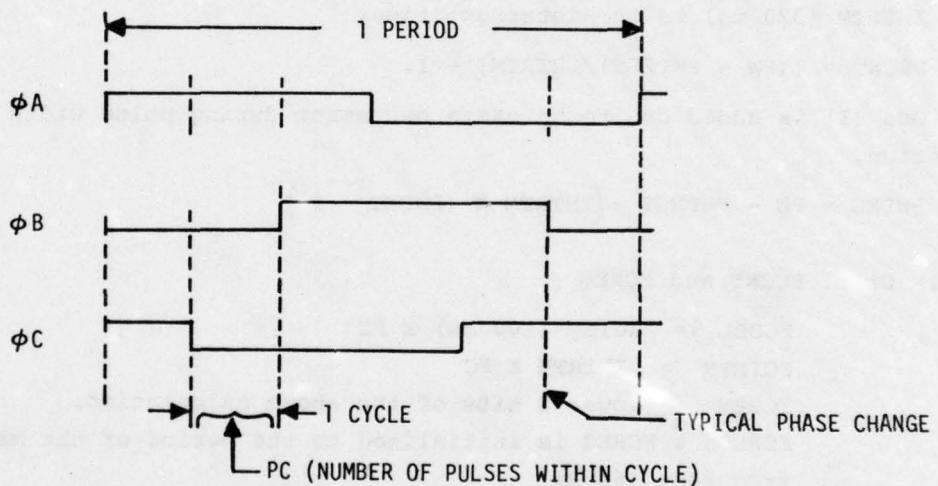
4.2.7 Adj FCREG, PC, PW

Frequency register (FCREGH and FCREG), number pulses per cycle (PC) and pulse width (PW) are generated as a function of frequency command (Fc) per the following table:

Fc	FCREGH & FCREG	PC	PW (ms)
45 to 70	104 H	1 (80 _H)	2.55 (FF _H) ≈ MAX PW
30 to 45	82 H	2 (40 _H)	1.27 (7F _H) ≈ MAX PW/2
10 to 30	41 H	4 (20 _H)	0.63 (3F _H) ≈ MAX PW/4
Below 10	20 H	8 (10 _H)	0.31 (1F _H) ≈ MAX PW/8

↑
Hexidecimal

NOTE: Assuming the 4 PC mode, the processor will output four pulses of 0.63 ms to the motor between each phase change. There are six phase changes each complete period.



4.2.8 Slip Control

SLIPN is incremented every 10 ms as a function of slip (SCTAC). Pulse width (PW) is varied as a function SLIPN resulting in an effective reduction in output voltage. The object is to operate the motor near its ideal slip resulting in an improvement in power factor and a reduction in motor current. The square root (modified) of slip is calculated and multiplied by (PW). A modified square root routine is

used due to its simplicity and speed of calculation. An exact square root routine would take far too long to calculate and is not necessary.

The following indicates why the square root is used.

Output motor torque is a function of output voltage squared, ($T = k_1 V^2$). Output voltage is a function a pulse width, ($V = k_2 PW$). Therefore if we make pulse width a function of square root of slip ($PW = k_3 (Sc-TAC)^{1/2}$), then output motor torque will be a direct function of slip. $T = k (Sc-TAC)$

4.2.9 Adj PW for B+

Pulse width is adjusted for changes in B+ resulting in a constant output voltage to the motor independent of B+.

$$ADJ\ PW = \frac{100V}{B+} (PW) \text{ for } B+ \text{ between 100 to 128 Volts.}$$

4.2.10 Calc. PWCNT and PWREG

PWPRGT (50 ms) is the PW program time delay.

INTRTM (320 ms) is the interrupt time.

$$PWCNT = [(PW - PWPRGT)/INTRTM] + 1.$$

One (1) is added due to an extra decrement during pulse width generation.

$$PWREG = PW - PWPRGT - (INTRTM \times (PWCNT - 1))$$

4.2.11 Calc. FCCNT and FCPEG

$$FCDEL = \text{delay (100 ms)} \times FC$$

$$FCINTR = INTRTM \times FC$$

FCREM = lower 8 bits of the above calculation.

FCREGH & FCREG is initialized to the period of the maximum frequency (64 Hz)

$$FCCNT = [(FCREG - FCDEL)/FCINTR] - 1.$$

One (1) interrupt is always intentionally missed during inner loop operation, therefore the minus one (-1).

$$FCREG = FCREG - FCDEL - ((FCCNT + 1) \times FCINTR)$$

4.2.12 Test Output

One word of previously selected processor internal data is outputted via the PIA to a digital to analog (D/A) module. Used for evaluating algorithm or system performance.

4.2.13 Transfer Information

Outer loop information is transferred inner loop to an inbetween transfer state. Transfer may occur only immediately following a return from inner loop. This is accomplished by clearing "zero" and then testing for a change in "zero". The inner loop loads "zero" with #1. The purpose of this subroutine is to prevent an improper mix of new and old data being used by the inner loop.

4.2.14 Multiply Subroutine

Multiples fractions or a fraction and a whole number. The steps are:

- (1) The multiplicand is stored in A.
The multiplier is stored in B.
- (2) The program branches to the multiply subroutine.
- (3) The multiplicand is stored in memory (MLTCAN).
- (4) The multiplication routine runs.
- (5) The result develops in A.
- (6) Branch back to main program. (Result is an A.)

4.2.15 Divide Subroutine

Numerator < Denominator

The steps are:

- (1) The numerator is stored in B.
The denominator is stored in A.
- (2) The program branches to divide subroutine.
- (3) The denominator is stored in memory DENOM.
- (4) The divide routine runs.
- (5) The result develops in A.
- (6) Branch back to main program. (Result is A.)

4.2.16 GENPH - Generate Phase Pointer and Phase

The phase pointer (PH PONT) points to one of six phase combinations stored in memory. As the PH PONT increments (1, 2, 3, 4, 5, 6, 1, 2, etc.) the phase combinations sequentially pointed to in memory represent a 3-phase clockwise rotation.

The phase pointer is incremented or decremented each time it is necessary to make a phase change. This is determined by multiplying (PC) by 2 and adding the results to (R) remainder. Each time a carry results the phase pointer is incremented or decremented depending on the sign of the frequency command (Sc Sign).

4.2.17 Output Information

Phase and PW are outputted to interface electronics by way of the PIA.

SECTION 5

HARDWARE DESCRIPTION

The Motor Controller, as implemented, required 5 standard rack mounted p.c. cards, 3 power output stages and six SCR driver cards.

The five p.c. cards are as follows:

- A1 - TTY Interface
- A2 - Computer Card
- A3 - Input Card
- A5 - Output Interface Card
- A6 - Adapter and D/A Card

A detailed functional description of these cards follows.

5.1 TTY Interface, Card A1

This card contains provisions for reset, halt, single step operation, TTY interface, and overvoltage protection for the +5 volt supply. Since all but the overvoltage protection are essentially standard circuits taken from the Motorola "M6800 Microprocessor Applications Manual," or Motorola Engineering Note "100", only this circuit will be described here.

The overvoltage protection circuit utilizes a dual transistor (T_1) to compare Zener D3 voltage with the +5 volt supply output voltage. If the 5-volt output is above a predetermined level (approximately 6 volts), transistor T_2 is turned on applying the gate signal to SCRL. This signal causes SCRL to turn on thereby shorting the output of the 5-volt supply to ground through current limiting resistor R23 and a 5 Amp. fuse. The 5 Amp. fuse will then blow disconnecting the +5 volt supply from the +5 volt bus. At this point, the +5 volt supply, regardless of its peak output level, is unable to damage circuitry connected to the bus.

5.2 Computer Card A2

The computer card consists of the following:

1. One (1) P.I.A. (A4) - MC 6820
2. One (1) MCM 6810 (A3) (128 x 8) Static RAM
3. One (1) MCM 6830L7 (A2 (1024 x 8) MIKBUG ROM
4. One (1) MC6871A (A7) two phase 1MHz Clock
5. Eight (8) 2102-1 (1024 x 1) Static RAMS
6. Two (2) DM 8097 Tri-State Gates
7. One (1) MC 6800 (A8) Microprocessing Unit
8. One (1) MC 6820 P.I.A. (A9) for Input/Output
9. One (1) 74LS138 Address Decoder (A1)
10. Two (2) Level Shifting buffers for "Register B" Output of the I/O P.I.A. (7417) (A6), (A10)
11. Two (2) 7400 Quad Gates for miscellaneous logic operations (A11), (A22).

} Teletype Interface

} (1024 x 8) Static RAM

It was found that the only problems encountered in implementing this card were the normal problems encountered when using unfamiliar circuits for the first time (e.g., misinterpretation of data sheet, polarity reversals, etc.). These problems were very quickly overcome and do not warrant discussion.

5.3 Input Card A3

Proper implementation of the algorithm developed for micro-computer control of the motor controller requires the continuous sampling of a number of variables such as: motor speed command, tachometer output, etc. Since all of these are most readily sensed as analog quantities, it remains for the controller to convert these quantities into a form suitable for communication with the microprocessor.

Communication with the MPU is provided by the Data Translation Inc., Model DT-830, 8-channel data acquisition module. This module consists of an 8-channel analog multiplexer, a sample and hold circuit; a 10-bit bipolar A to D converter and associated control circuitry. A 3-bit address is provided to the MUX address input line by the MPU PIA output lines PB4, 5, and 6. By means of this, it is possible to select any one of the eight input channels available; only five channels are

required for this application. A clock pulse derived from CB2 by card A6, Adapter and D/A card, ensures that the address information presented to the MUX has stopped changing before a convert command is issued to the DT-830.

When a convert command is received, the sample and hold circuit, which has been tracking the input analog signal, holds the input signal until the conversion is completed.

The 10-bit successive approximation A/D converter makes the data available through tri-state output gates. The MSB (pin 62) is connected to PB7 while the next 8 bits are sampled by the data bus (PA0 - PA7); the LSB is not used in this application. Provision is also made on the card to enable the application of external calibration voltages to the DT-830. This permits calibration of the unit without removing it from card A3.

The board also contains signal conditioning circuitry for the tachometer output voltage and a means of generating a speed command input to the MPU.

All that is needed for the tachometer signal conditioning is an operational amplifier having low pass filtering in both the input and output circuits and a dc gain which provides the proper scaling factor between tachometer output and A/D input.

The speed command is derived from the ± 15 -volt supplies using a potentiometer across two 5.1 volt Zener diodes thereby providing a continuously varying voltage from +5 to -5 volts. This voltage is applied to a rate limiting circuit which prevents application of a step change to the controller input. Provision has also been made to apply a step of either +5 volts or -5 volts by setting the pot to 0 volt and activating a three-position toggle switch.

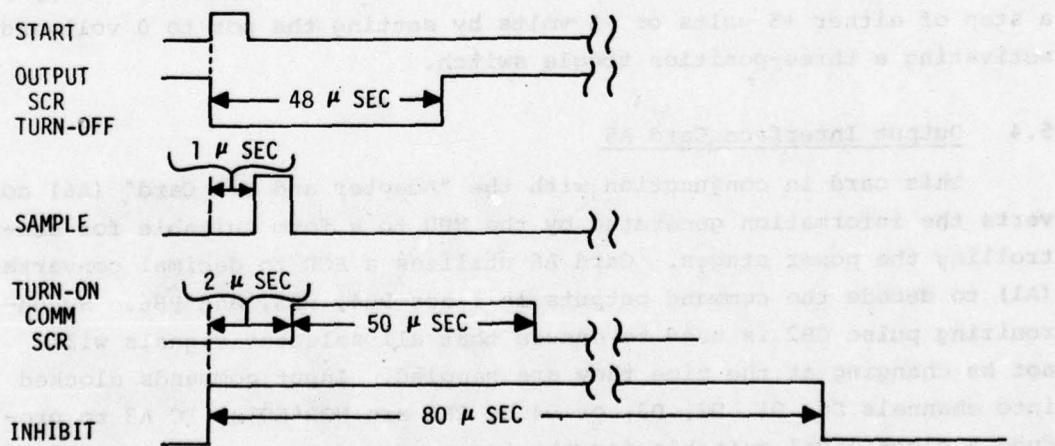
5.4 Output Interface Card A5

This card in conjunction with the "Adapter and D/A Card" (A6) converts the information generated by the MPU to a form suitable for controlling the power stages. Card A6 utilizes a BCD to decimal converter (A1) to decode the command outputs on lines PB4, PB5, and PB6. Synchronizing pulse CB2 is used to ensure that all selected signals will not be changing at the time they are sampled. Input commands clocked into channels D0, D1, D2, D3, or D4 by CB2 are NOR'ED by IC A3 to produce a clock level suitable for the input card data acquisition module.

Phase output sampling is selected by channel D5 which also produces the clock input necessary for the output card (A5). Calling up channels D6 and D7 of IC A1, permits the sampling of the data on lines PB0 through PB3 by the AD 7520 8-bit D/A converter IC (A7). With D7 sampling the higher order bits and D6 the lower order bits. ϕ A, ϕ B, ϕ C information is operated on by the 8-channel data selector A9 to produce the ex or signal needed by output card A5. All of the signals produced by P.C. card A6 with the exception of the D/A output and the input clock are routed to Output Card A5 where they are used to generate the signals required by the power stage and power stage SCR drivers.

On card A5 the ϕ A, ϕ B, ϕ C, PW and ex or information are combined by the quad and/or select gate (IC A9) to produce the waveform desired for the SCR gate drivers. This is accomplished by placing ϕ A, ϕ B, and ϕ C information on the A inputs and the ex or information on the B inputs. The selection of the A or B inputs is then made to depend on the absence or presence of a "1" on the PW input. There is also a provision at this point to have a peak current limit condition or the "Clear" position of the "Clear-run" switch force the IC to select B inputs. At the same time the B inputs are forced to a "1" level.

The discussion up to this point has covered the generation of the proper waveforms to correctly operate the Power stage SCR drivers. If these stages are to function properly, the timing relationship between sample time, turn off of the output SCR's, and turn on of the commutation SCR's must be maintained. The correct timing is shown below:



Each time the clock input of Card A5 goes high, a 1 level is presented to the D Input of the MC 14015 dual 4-bit shift register (S.R.). This level is clocked into the S.R. by the 1 MHz clock Q2T. The inhibit pulse then removes the level to prevent multiple pulses from being loaded. Successive Q2T pulses shift this input pulse through the register and provide the output pulses shown above with the aid of three (3) one shot multivibrators. The main and commutate SCR pulses are named with the ϕ A, ϕ B,

5.5 SCR Driver Card

Each driver card consists of a main gate drive circuit and a commutate gate drive circuit. The commutate command is on the order of 50 milliseconds and, therefore, a pulse transformer with single ended drive is quite suitable. The main gate drive commands are much longer, therefore, a push-pull 25 KHz transformer coupled circuit with dc rectification in the secondary is utilized. Both circuits provide for gate current overdrive at turn-on. This insures higher di/dt SCR capability.

5.6 Interrupt Generation

A module-N-divider in conjunction with a one-shot generate a series of interrupt pulses at 320 milliseconds intervals. The timing of these pulses is reset at the start of each output pulse width. The function is implemented on card A1 and A6.

5.7 Power Stages

The McMurray Inverter was chosen as the power stage for this controller. The circuit is the classic McMurray circuit with the addition of a few added components dictated by real world practicalities.

Since the McMurray circuit has been described many times in the current technical literature, a description of its operation will not be included here.

The most noticeable deviation from the classic, is the inclusion of 40 μ H inductors in the power stage (refer to the accompanying schematic). These pulsus the RC snubbers across the individual SCR's were added to limit the reapplied dv/dt.

SECTION 6

SYSTEM SIMULATION

6.1 General Description

A FORTRAN model of the controller, motor and load was generated and a simulation was conducted on The Charles Stark Draper Laboratory Amdahl computer. The simulation helped verify the overall concept and led to refinements in the control algorithm. The simulation uncovered a basic flaw in a control function (IFOLD) and aided in finding a solution.

6.2 Model Organization

- (1) CNTRL Subroutine - Models the control algorithm which exists in the microprocessor. The prime inputs are speed command (SC), Motor speed (TAC) and motor current (I). The prime outputs are frequency command (F) and Pulse width (PW).
- (2) HRMNIC Subroutine - Determines the Fourier harmonics of the controller's output waveform which is delivered to the motor. The prime inputs are (F) and (PW) from which the shape of the waveform is determined and battery voltage (VB). The prime outputs are the Fourier voltage coefficients V1, V5, V7, etc.
- (3) MOTORZ Subroutine - Models the motor. The prime inputs are RMS voltage (V), harmonic number (N), frequency (F) and motor speed (STAC). The prime outputs are RMS motor torque (TQ), motor input current (I1), rotor current (I2), power factor (pf) and rotor loss (ROTL).
- (4) MOTOR Subroutine - Passes the Fourier voltage coefficients (V1, V5, V7, etc.) along with harmonic number, frequency and motor speed to the MOTORZ subroutine. Sums the resulting harmonic torques, currents, and rotor losses.

- (5) LOADG Subroutine - Models a generator load. The prime inputs are motor torque (TORQ) and motor-load speed (RDTAC). The resulting output is the updated motor-load speed (RDTAC).
- (6) LOOP Subroutine - Sequences the program through the subroutines and outputs the results to data files or the printer.
- (7) MAIN Subroutine -
 - (a) Defines system parameters
 - (b) Establishes test conditions
 - (c) Calls loop a prescribed number of times

6.3 FORTRAN Model Program

Printout sheets for the FORTRAN model program are given in Appendix A.

6.4 Simulation Results

- (1) Transient speed control
 - (a) The motor current is less, the motor torque is higher and the response of the motor faster by utilizing transient speed control versus allowing step speed command to be applied to the motor. The reduction in motor current is on the order of 2.5 to 1.
 - (b) DLTAUP = 7.5 Hz proved to be a good choice.
DLTADW = 3 Hz is a reasonable compromise. Initially it had been set to 7.5 Hz which did not reduce reversing currents sufficiently.
 - (c) TLL2 HYSTERESIS lowered to 5.5 and 5.0
 - (d) Motor starting: FC = BNDRYL + TAC.
Had been FC = BNDRYL.
- (2) Multiple pulse per cycle boundaries - The frequency boundaries of the various pulse modes were adjusted to minimize motor current harmonics. The new boundaries are:

1 P/C	45 to 60 Hz
2 P/C	30 to 45 Hz
4 P/C	10 to 30 Hz
8 P/C	Below 10 Hz

- (3) Slip Control - The basic approach functioned as designed. Some improvement in motor performance if PW was incrementally reduced in a delayed manner as a function of the ratio of slip/ideal slip.
- (4) Current Foldback of Speed Command IFOLD - Significant changes were necessary to attain proper operation of this control function. Limiting speed command (SC) to the ratio of $\frac{IAVE - ILIM}{ILIM - IMAX} \times MAX SC$ was found to be a sound concept. The simulation showed that large motor currents will occur unless the value of SC limit is incremented to its final value in a delayed fashion. Essentially what had been initially programmed was an unstable feedback loop and, therefore, compensation was required.

6.5 Simulation Curves

- (1) Model Block Diagram (See Appendix A)
- (2) Step Functions - speed, motor torque, and motor current are plotted vs time.

SF #1 SC = 0 to 60 Hz No Controller

SC = +60 Hz to -60 Hz No Controller

SF #2 SC = 0 to 60 Hz Controller DLTUP = 7.5 Hz

SC = +60 Hz to -60 Hz Controller DLTDW = 7.5 Hz

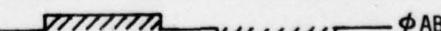
DLTUP = 7.5 Hz

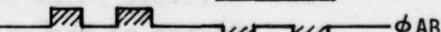
SF #6 SC = 0 to 60 Hz Controller DLTDW = 3 Hz

SC = +60 Hz to -60 Hz Controller FINAL PARAMETER
CONFIGURATION

- (3) FOURIER Coefficients vs Percent Modulation

(a) Percent Modulation = (total pulse width/(2/3) period)* 100%

100% Modulation waveform 

50% Modulation waveform 

(b) 1st (V1) through 25th (V25) harmonic are plotted.

(c) Curves of 1, 2, 4 and 8 pulse modes are presented.

- (4) Percent Harmonic at Various Pulse Modes vs. Frequency - Curves represent what the percent harmonic content would be if a fixed pulse mode were maintained over the full speed range.

- (5) MOTOR Z Curves - Motor torque, current and power factor vs motor speed.
 - (a) Line frequency = 60 Hz
 - (b) Line frequency = 30 Hz
 - (c) Line frequency = 15 Hz
- (6) ILIMIT
 - (a) Motor speed, torque and current vs. time.
 - (b) Test description
 - (1) Motor reverse from -60 Hz to 60 Hz @ T = 0
 - (2) Load torque increased step wise to exceed I limit @ T = 125
 - (3) Load torque increased further (time ramp) @ T = 300
 - (c) Controller configuration
 - (1) ILIMIT = .F. ILIMIT subroutine bypassed
 - (2) ILIMIT = 50 to 100 AMPS ILIMIT subroutine utilized
 - (d) Results
 - (1) Reversing current undisturbed by ILIMIT subroutine
 - (2) ILIMIT subroutine successfully limits motor current.
- (7) Slip Control
 - (a) Motor torque, volts, current and PF vs time.
 - (b) Test description
 - (1) Normal load torque @ T = 0
 - (2) Incrementally reduces load torque @ T = 25
 - (c) Controller configuration
 - (1) Slip Control = .F. - slip control subroutine bypassed.
 - (2) Slip Control = .T. - slip control subroutine utilized.

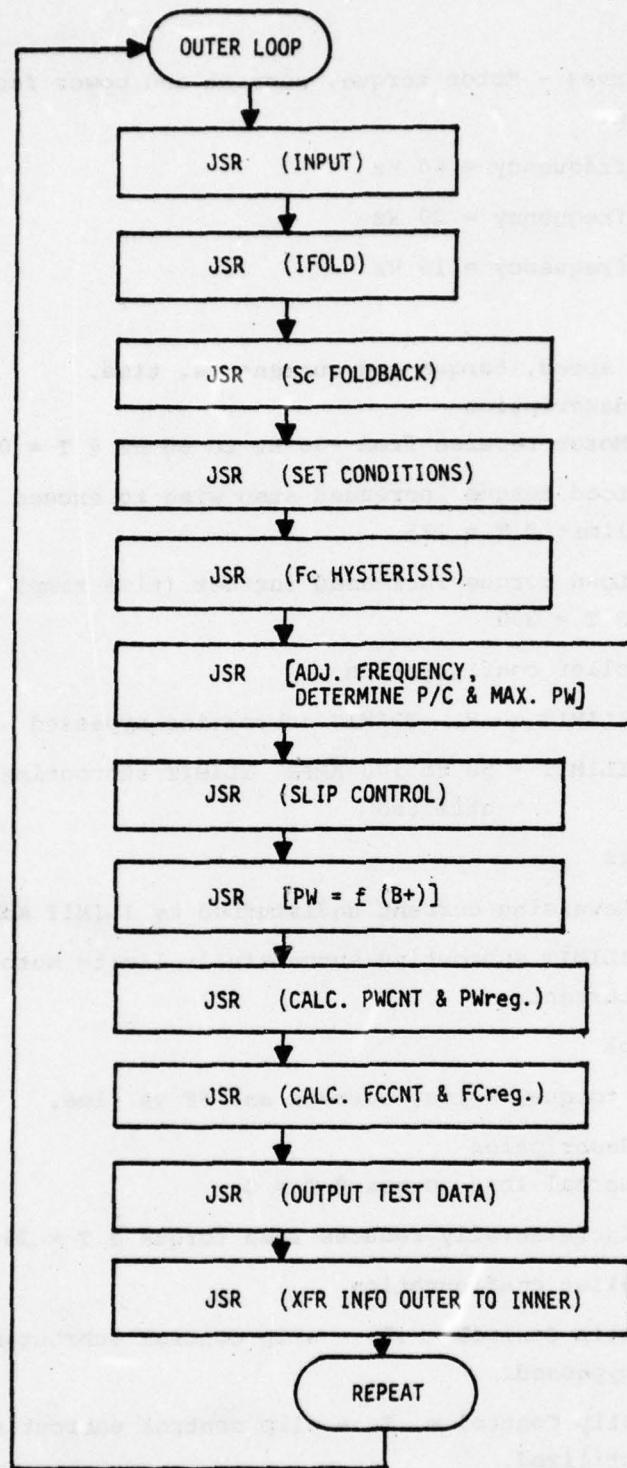


FIGURE 3. OUTER LOOP SUBROUTINE FLOW

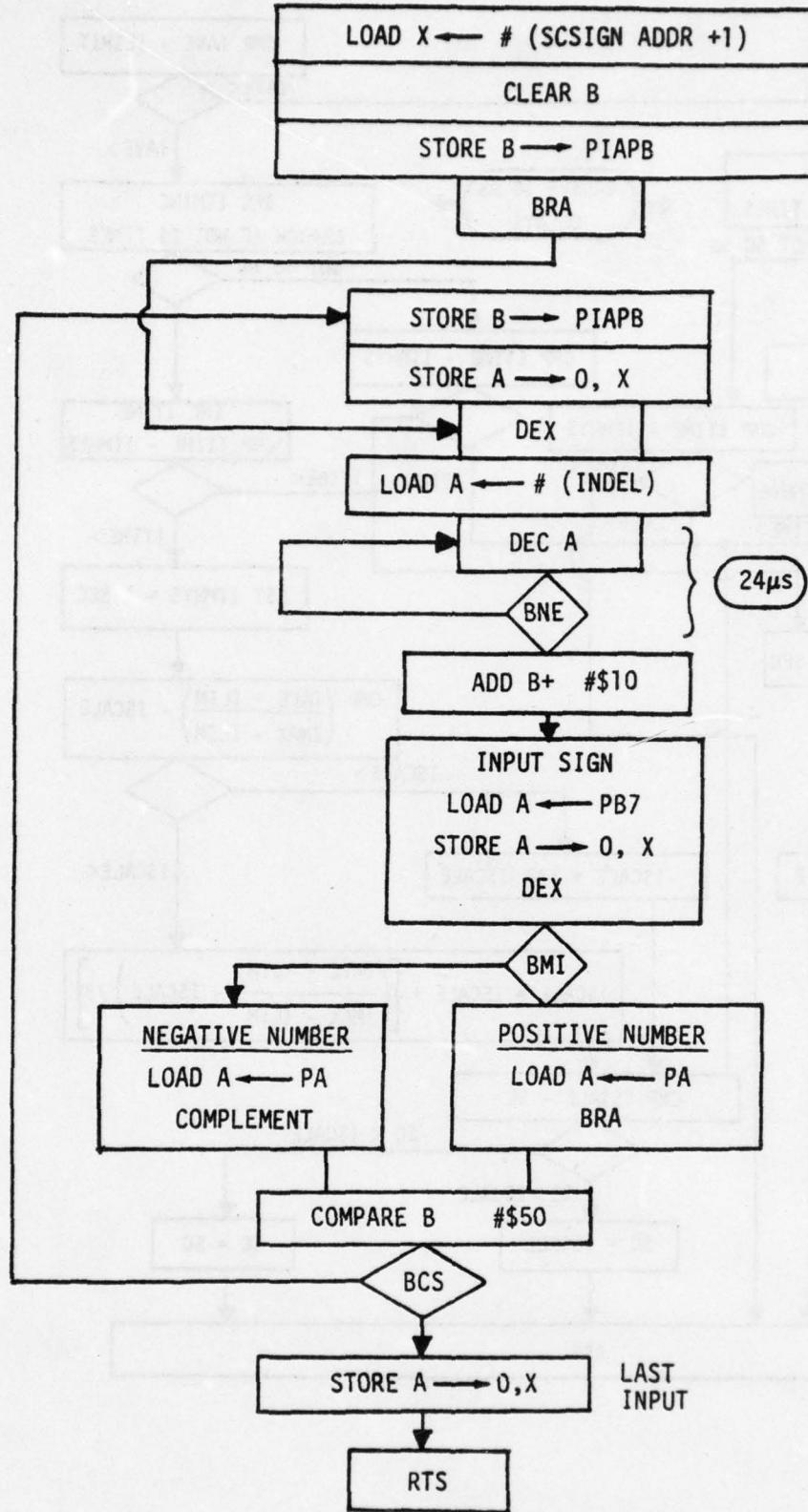


FIGURE 4. INPUT INFORMATION.

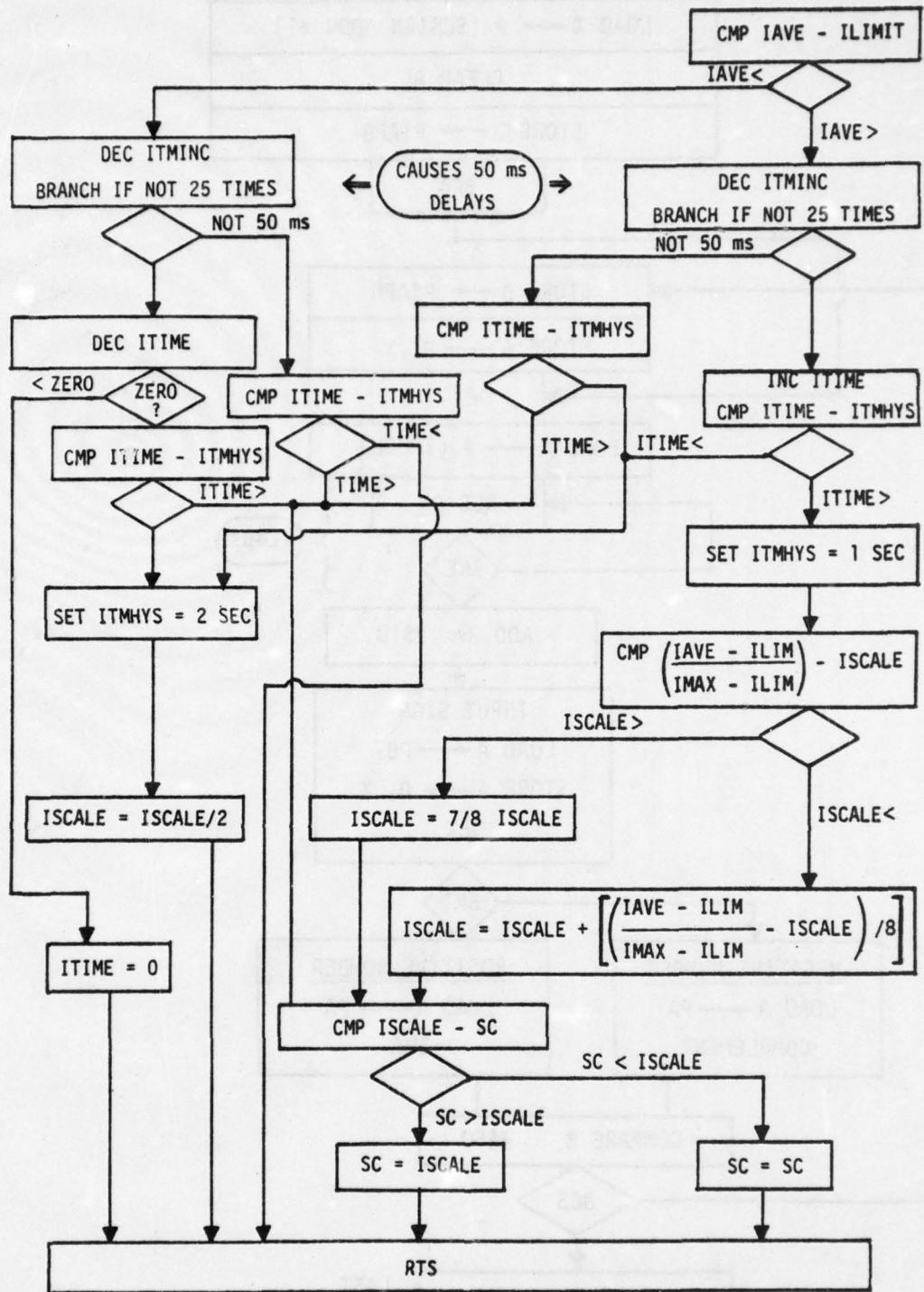
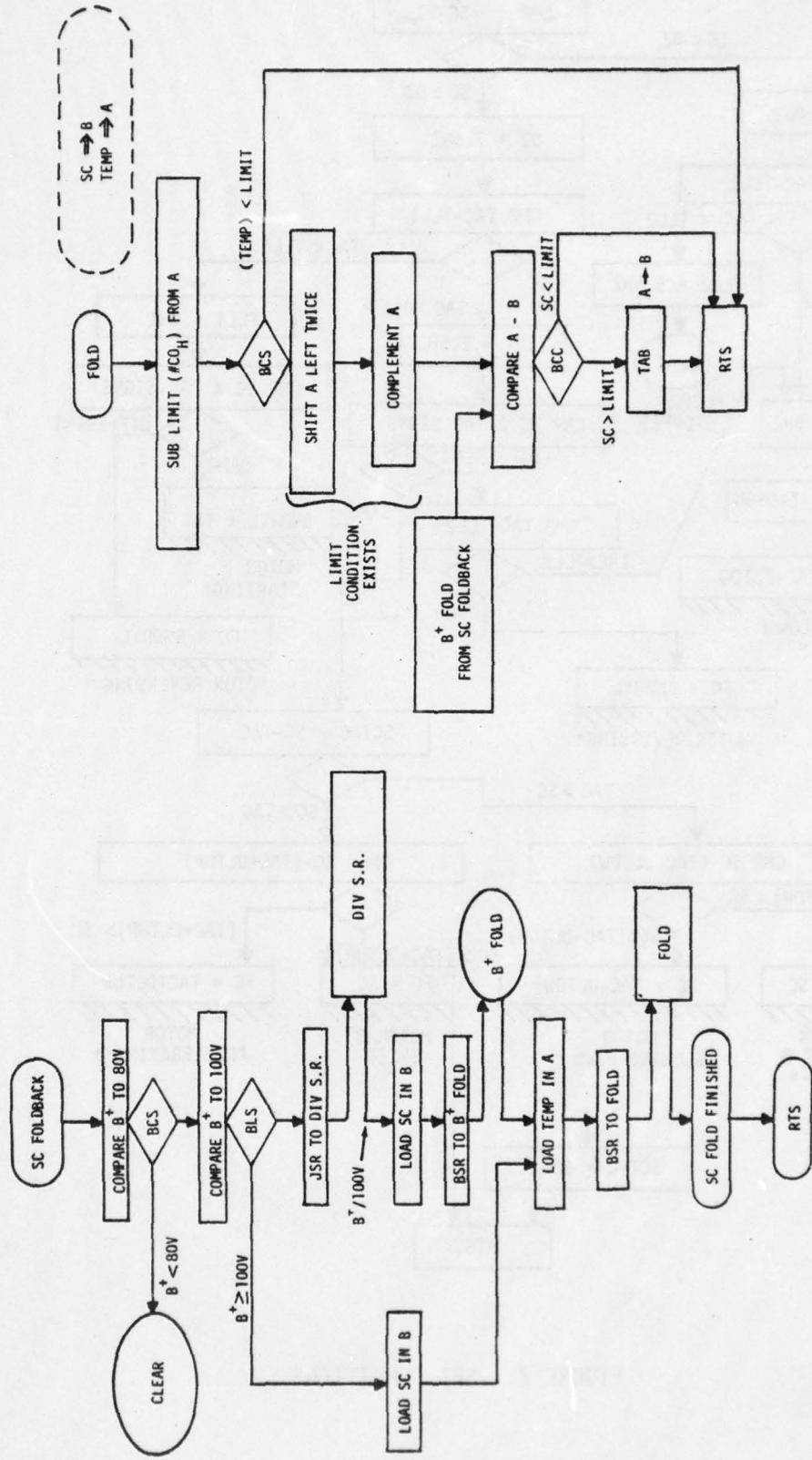


FIGURE 5. IFOLD.

FIGURE 6. SC FOLDBACK B^+ , IAVE, TEMP.



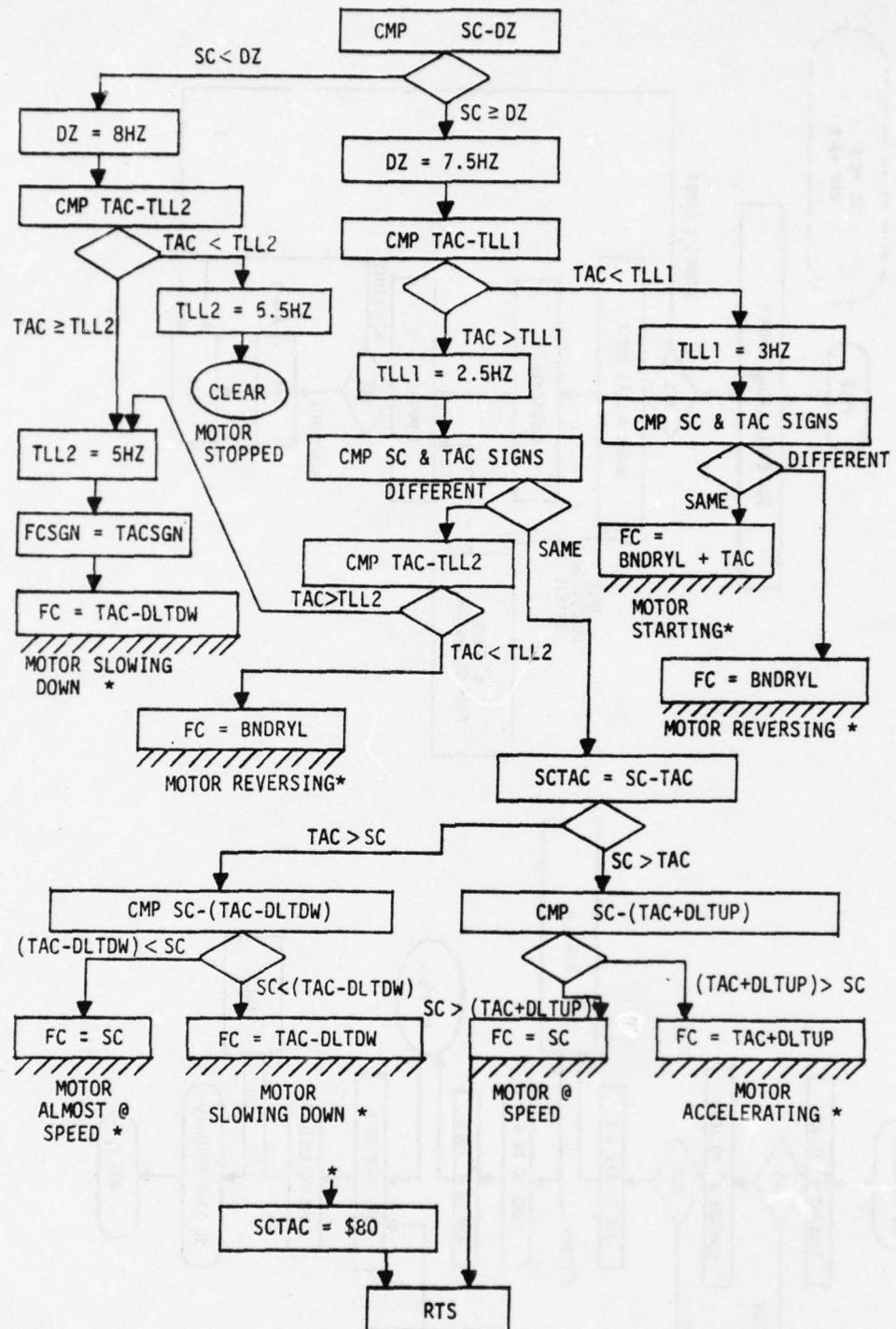


FIGURE 7. SET CONDITIONS.

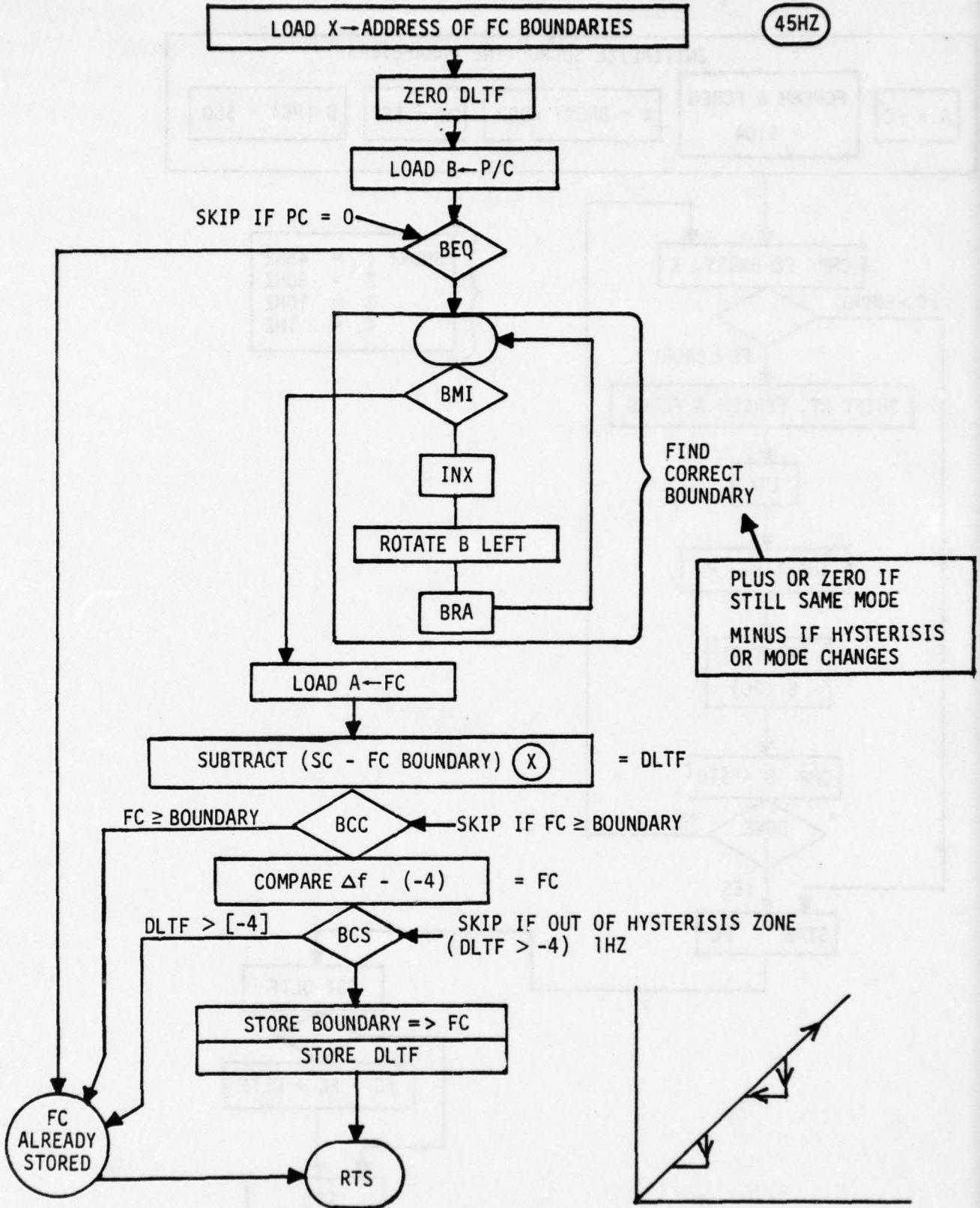


FIGURE 8. FC HYSTERESIS.

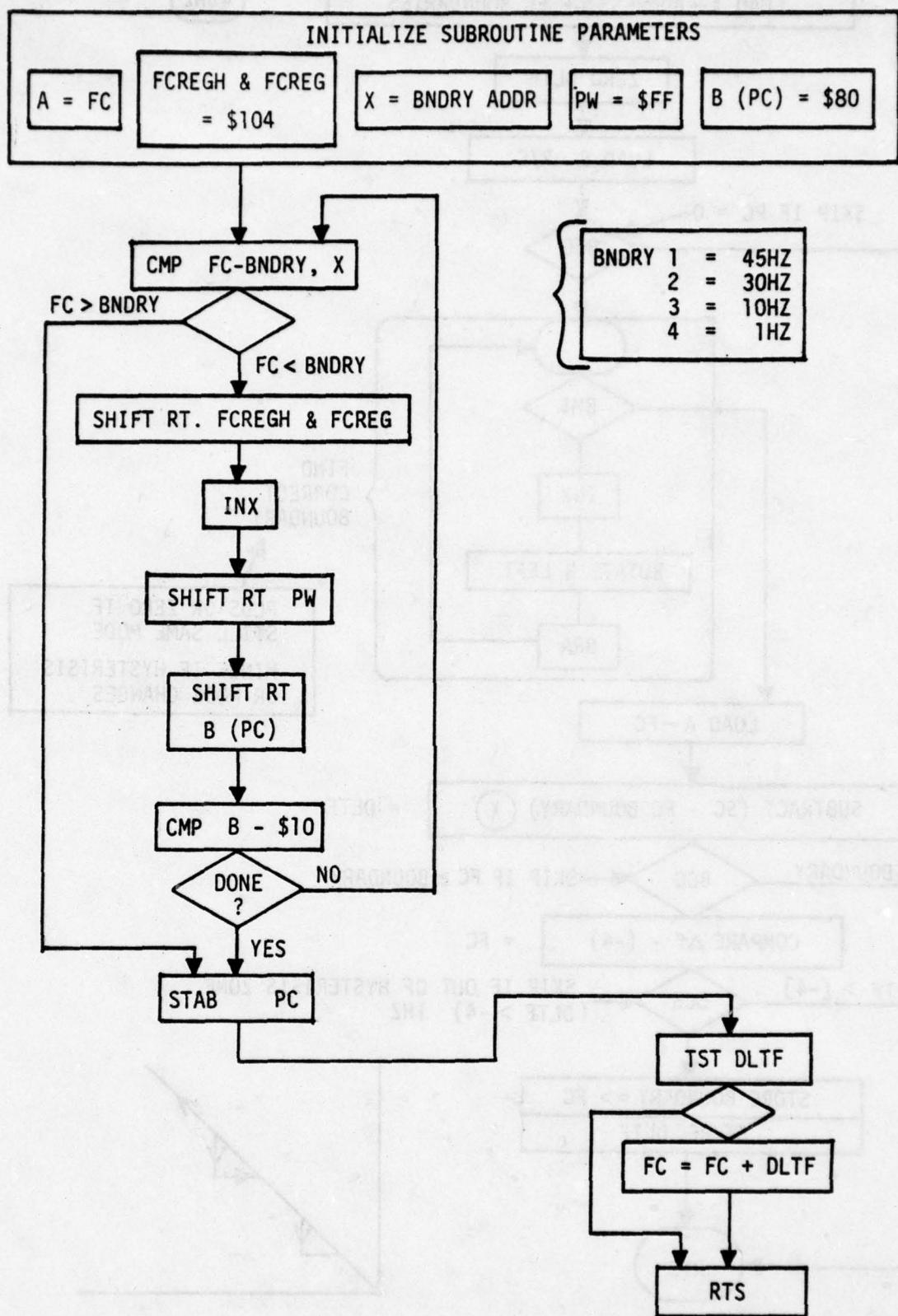


FIGURE 9. ADJ FCREG, DETERMINE PC & PW.

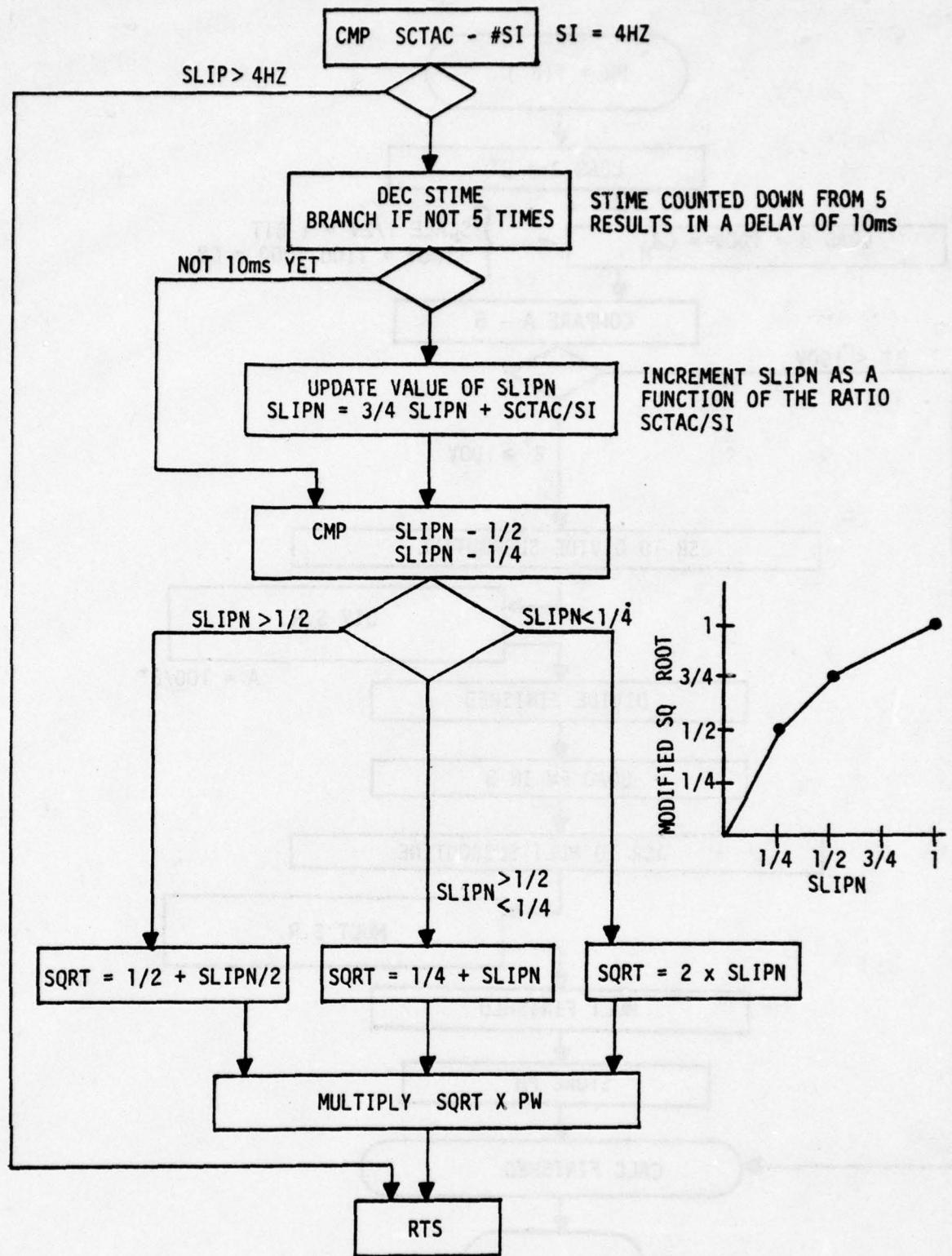


FIGURE 10. SLIPC.

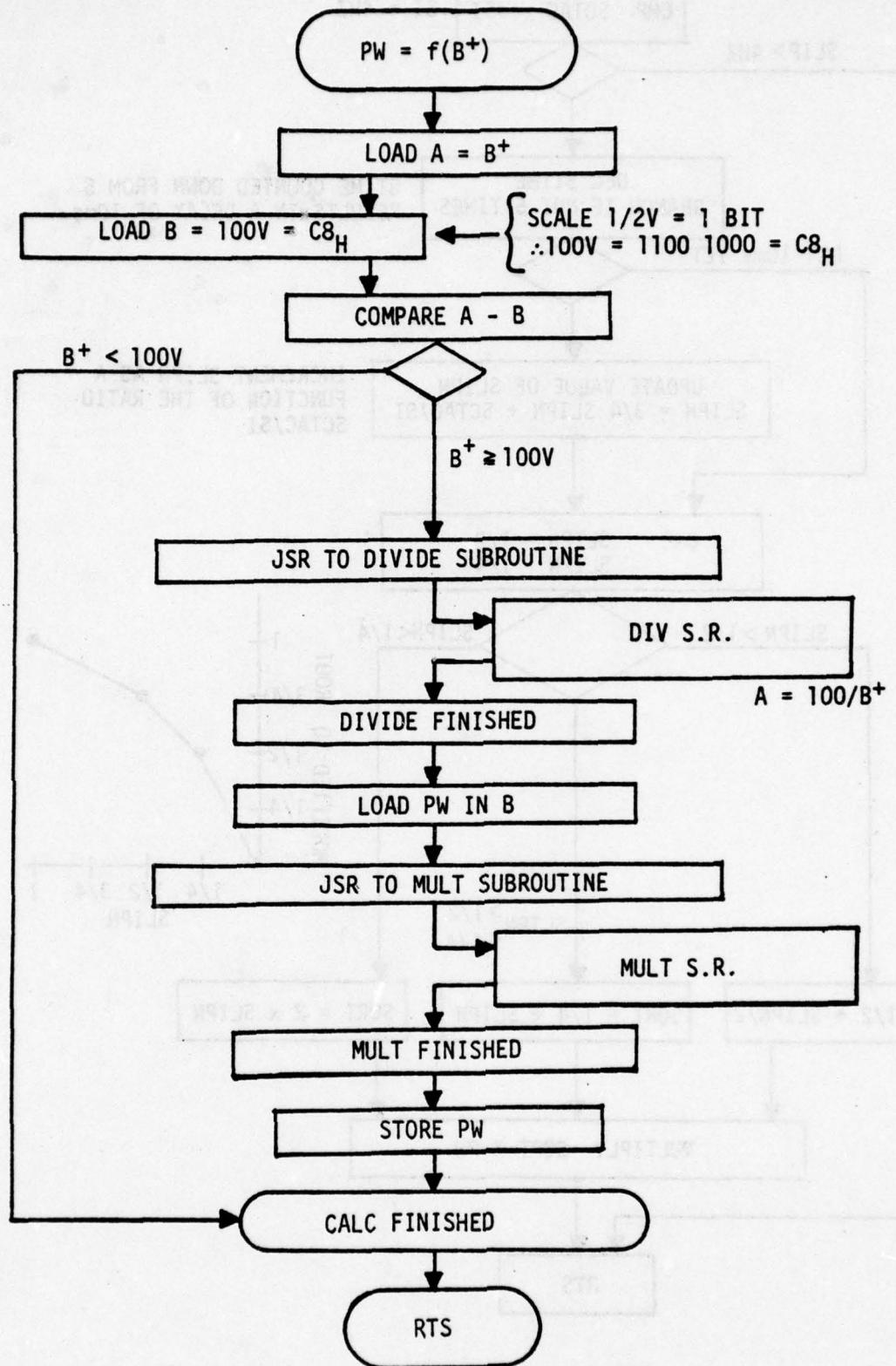


FIGURE 11. ADJUST PW FOR B^+ [100V TO 128V RANGE].

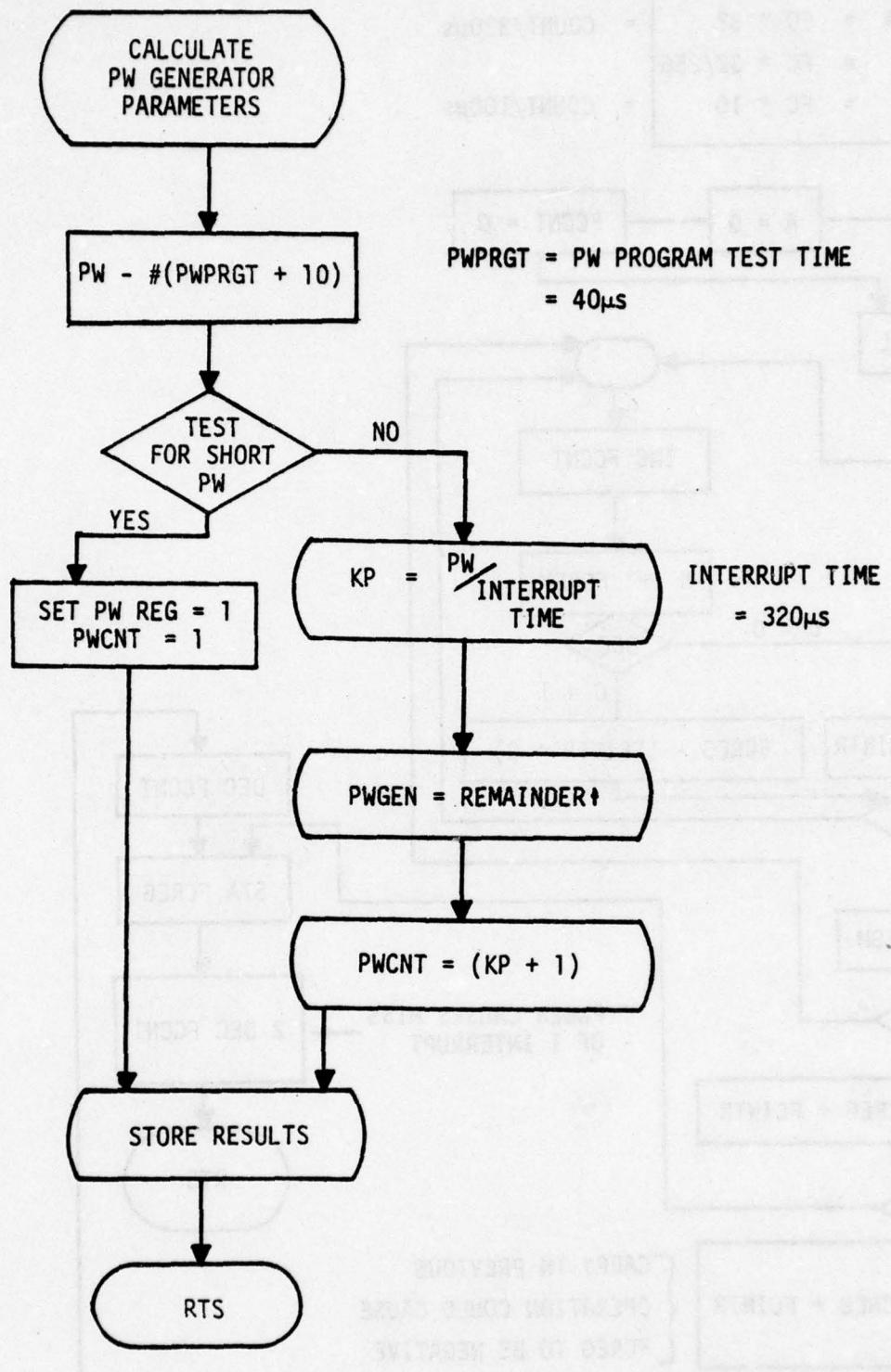


FIGURE 12. PWPT.

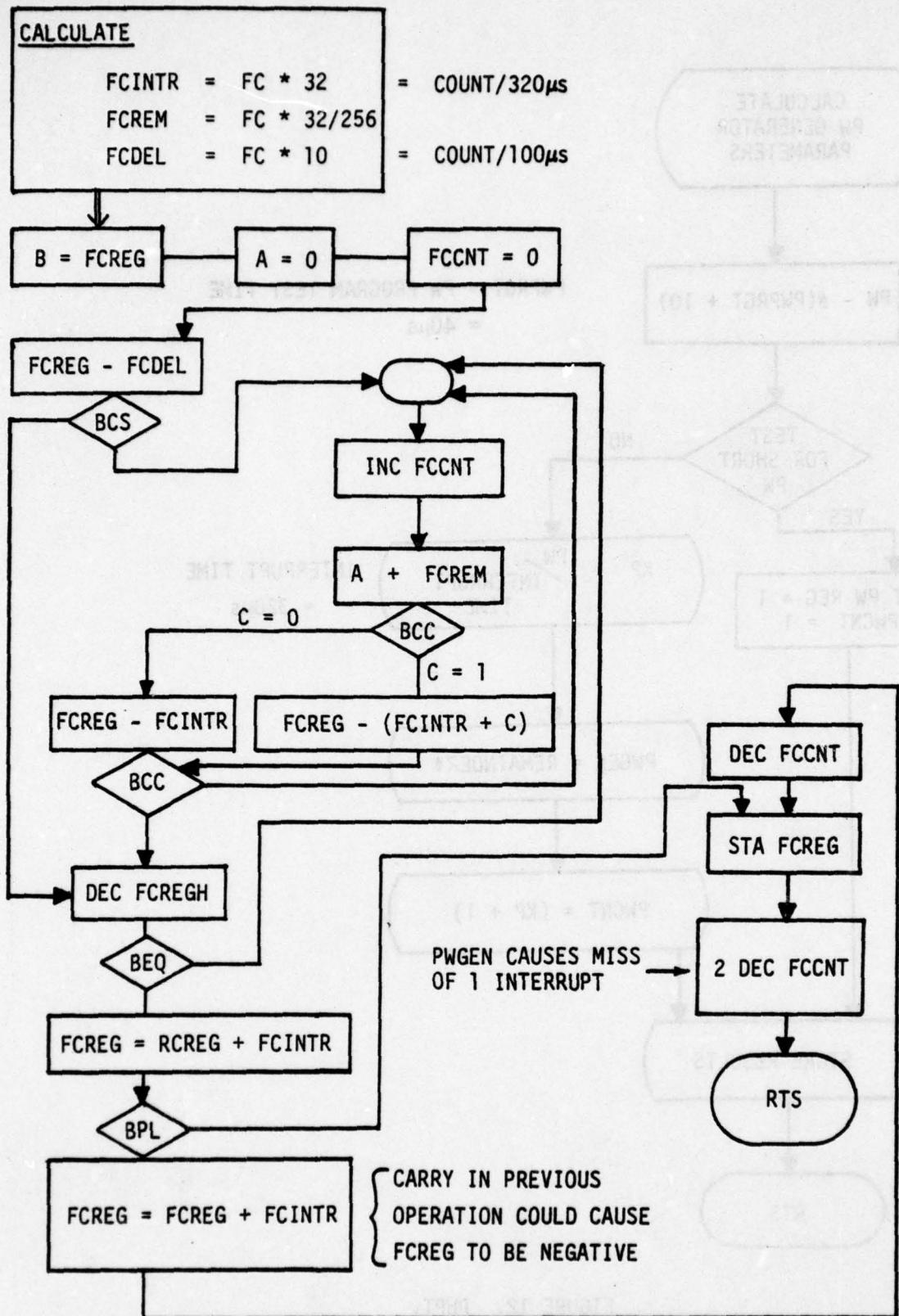


FIGURE 13. FC POINTER SR.

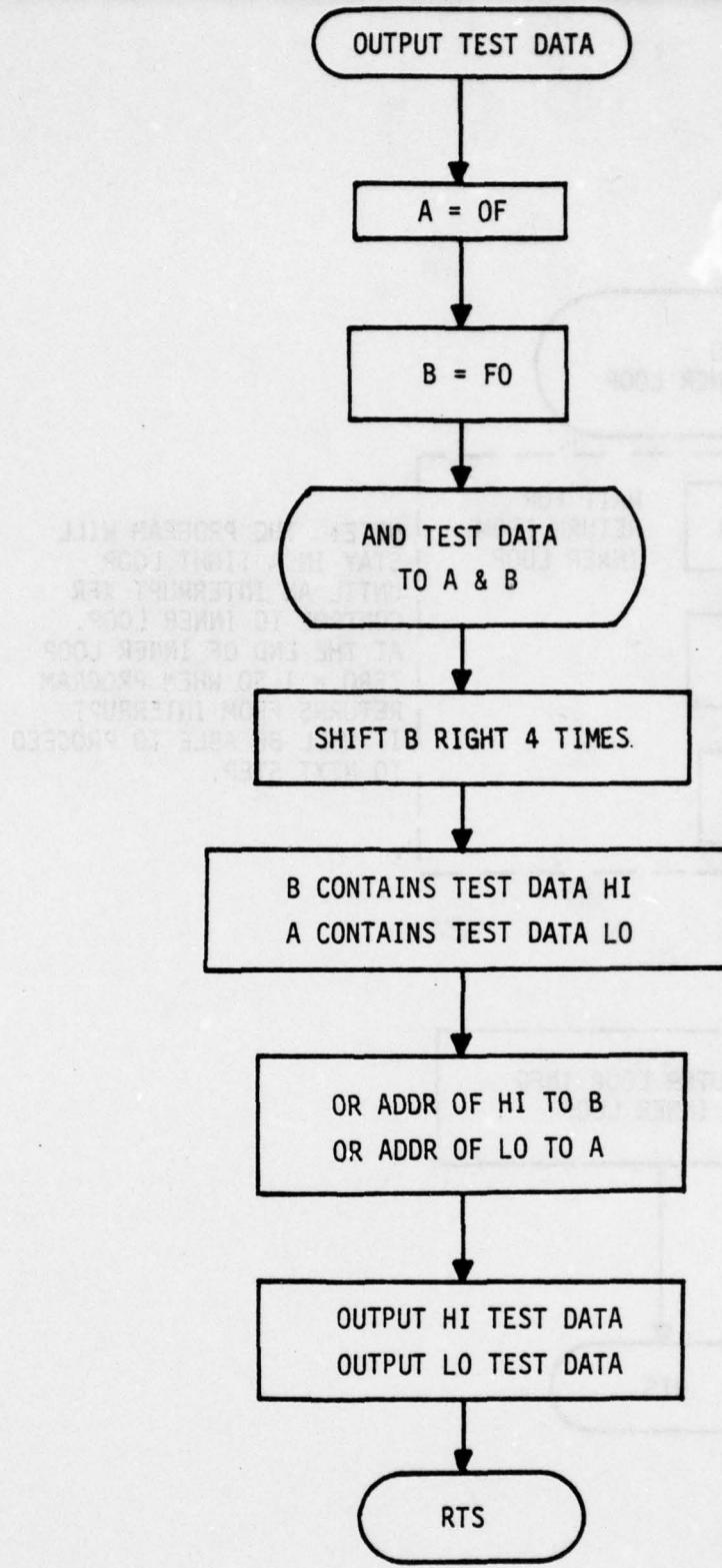


FIGURE 14. OUTPUT TEST DATA.

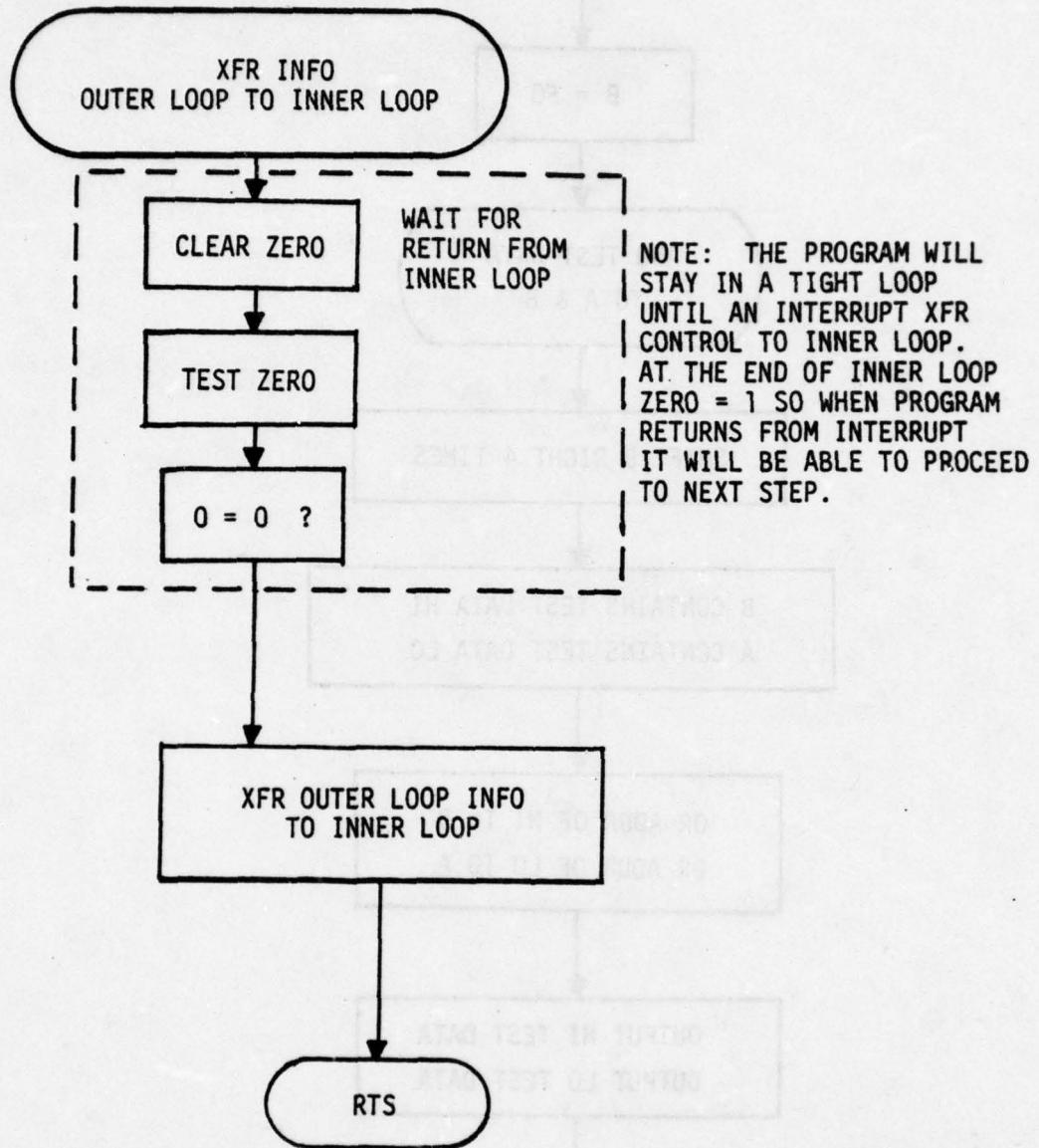


FIGURE 15. TRANSFER INFORMATION

- 1) NUMERATOR IN B
- 2) DENOMINATOR IN A
- 3) NUMERATOR < DENOMINATOR
- 4) STORE DENOMINATOR IN MEMORY
- 5) FORM RESULT IN A

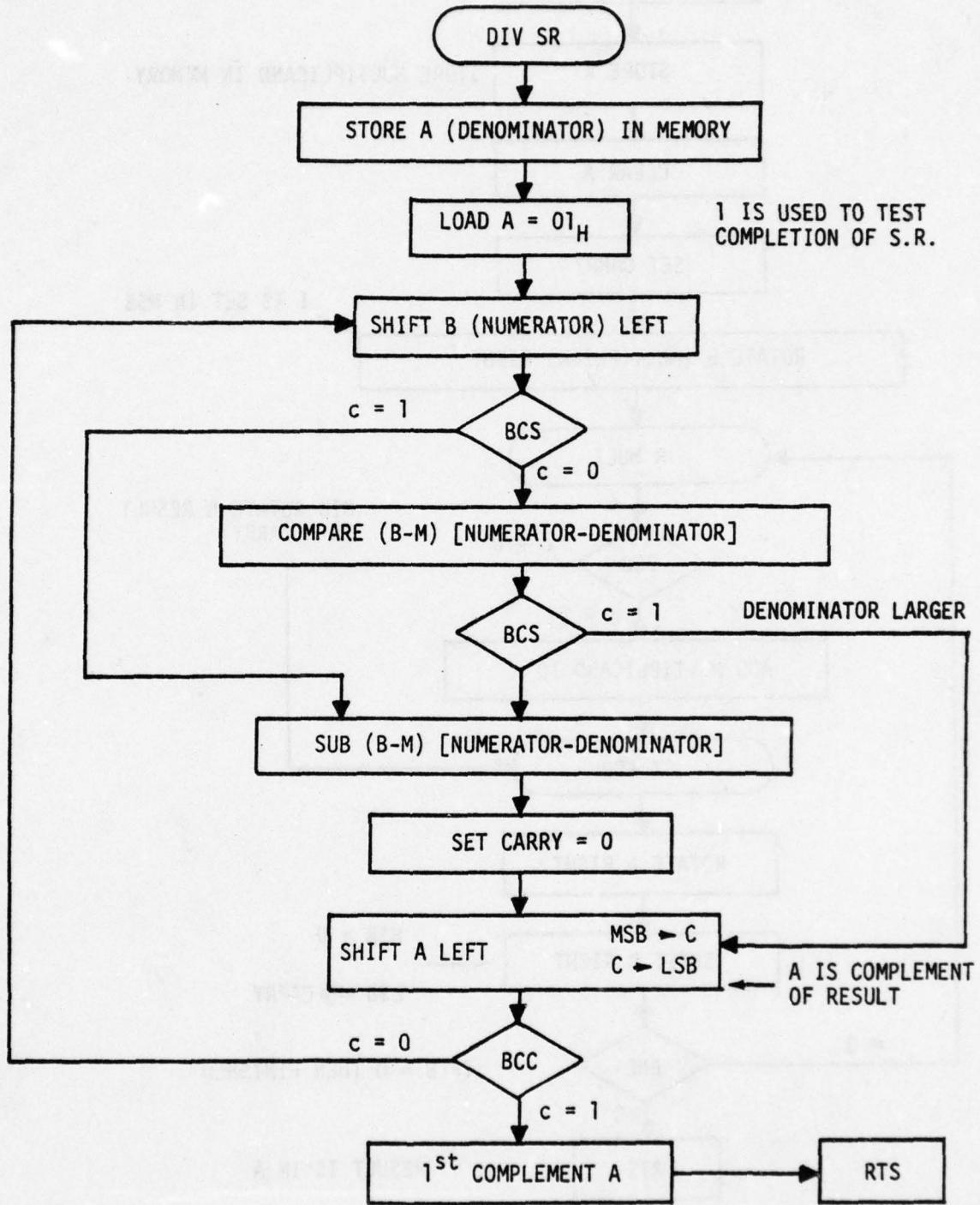


FIGURE 16. DIVIDE SUBROUTINE.

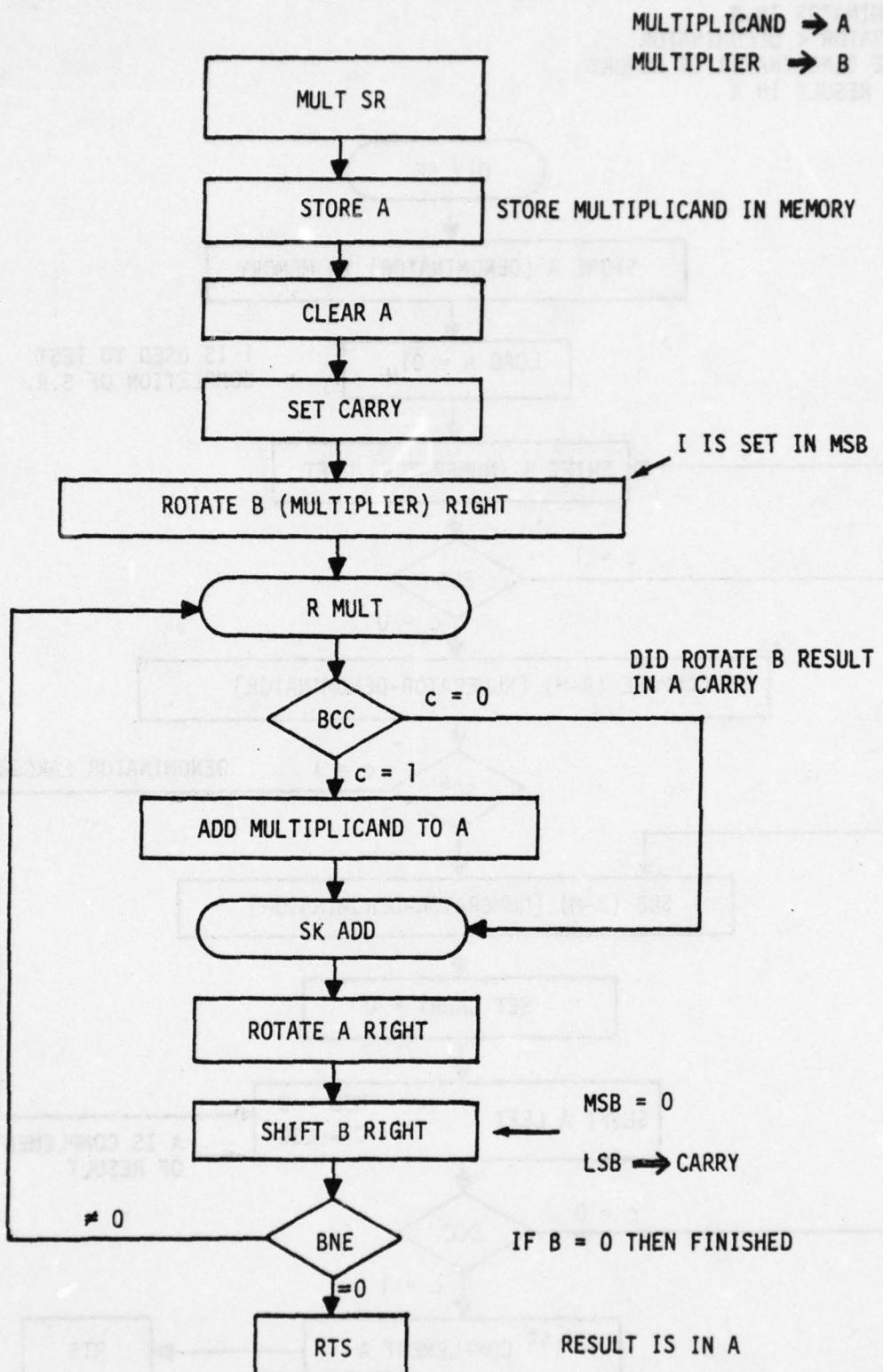


FIGURE 17. MULTIPLY SUBROUTINE.

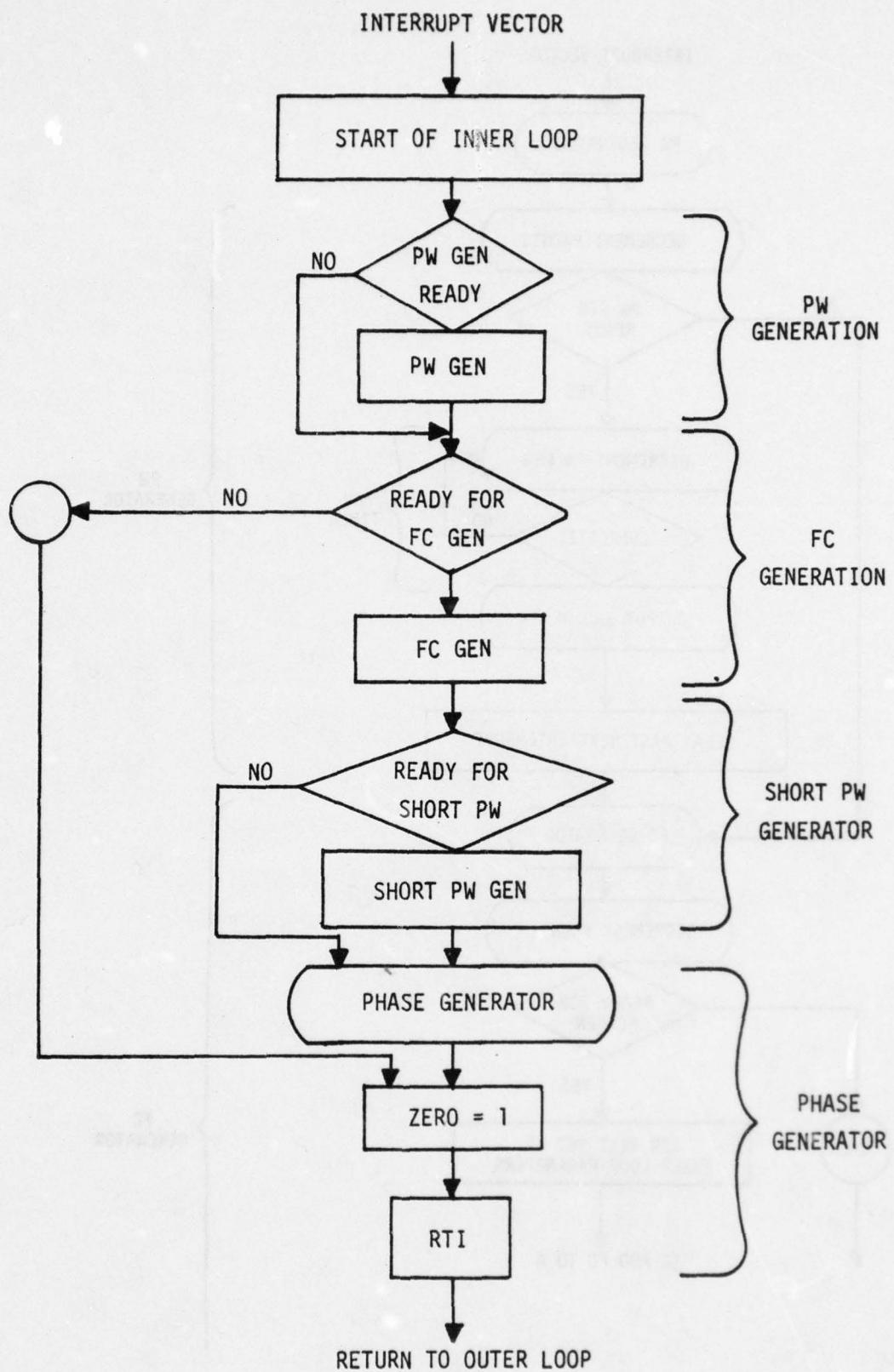


FIGURE 18. OVERVIEW OF INNER LOOP.

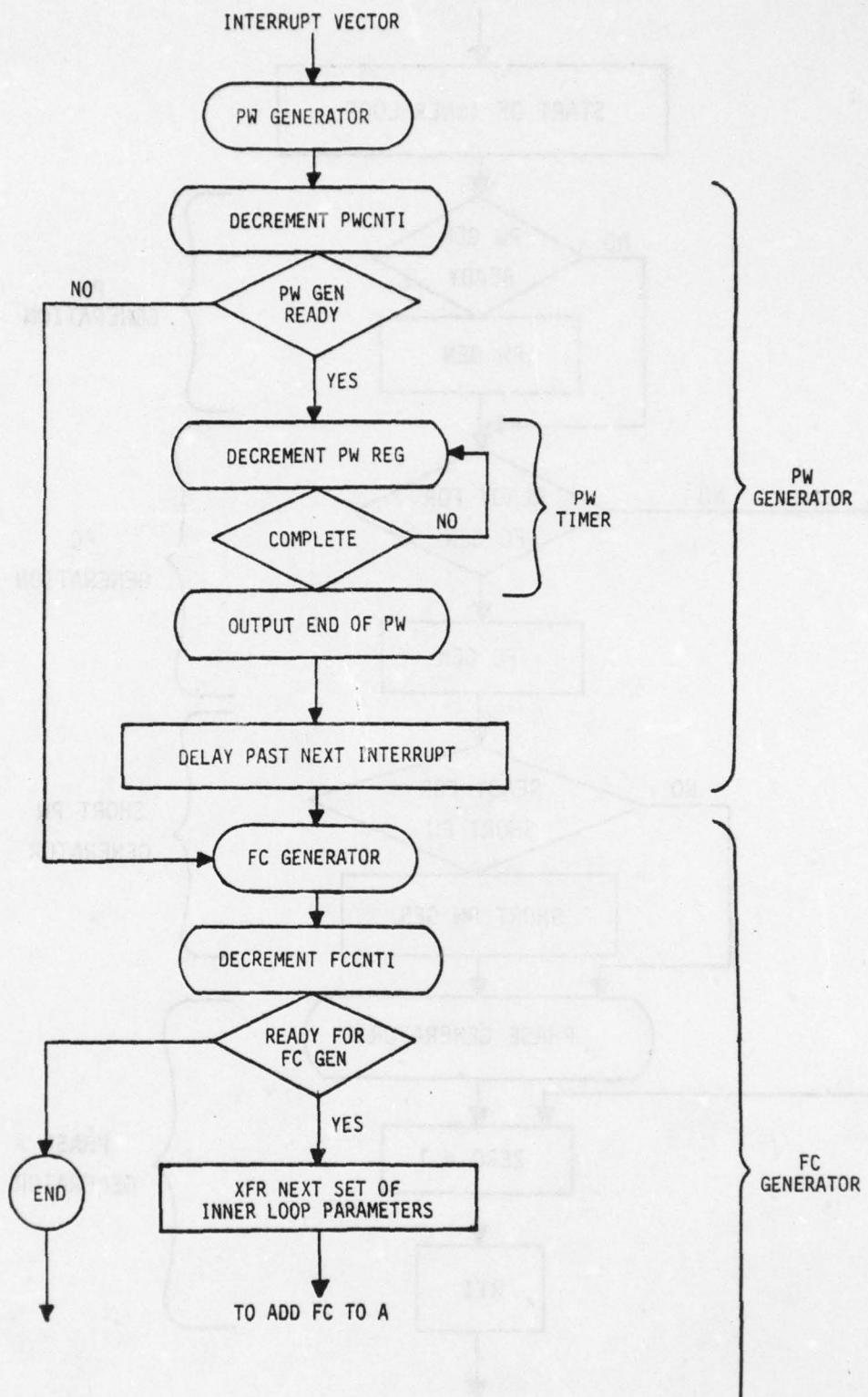


FIGURE 19. INNER LOOP 1.

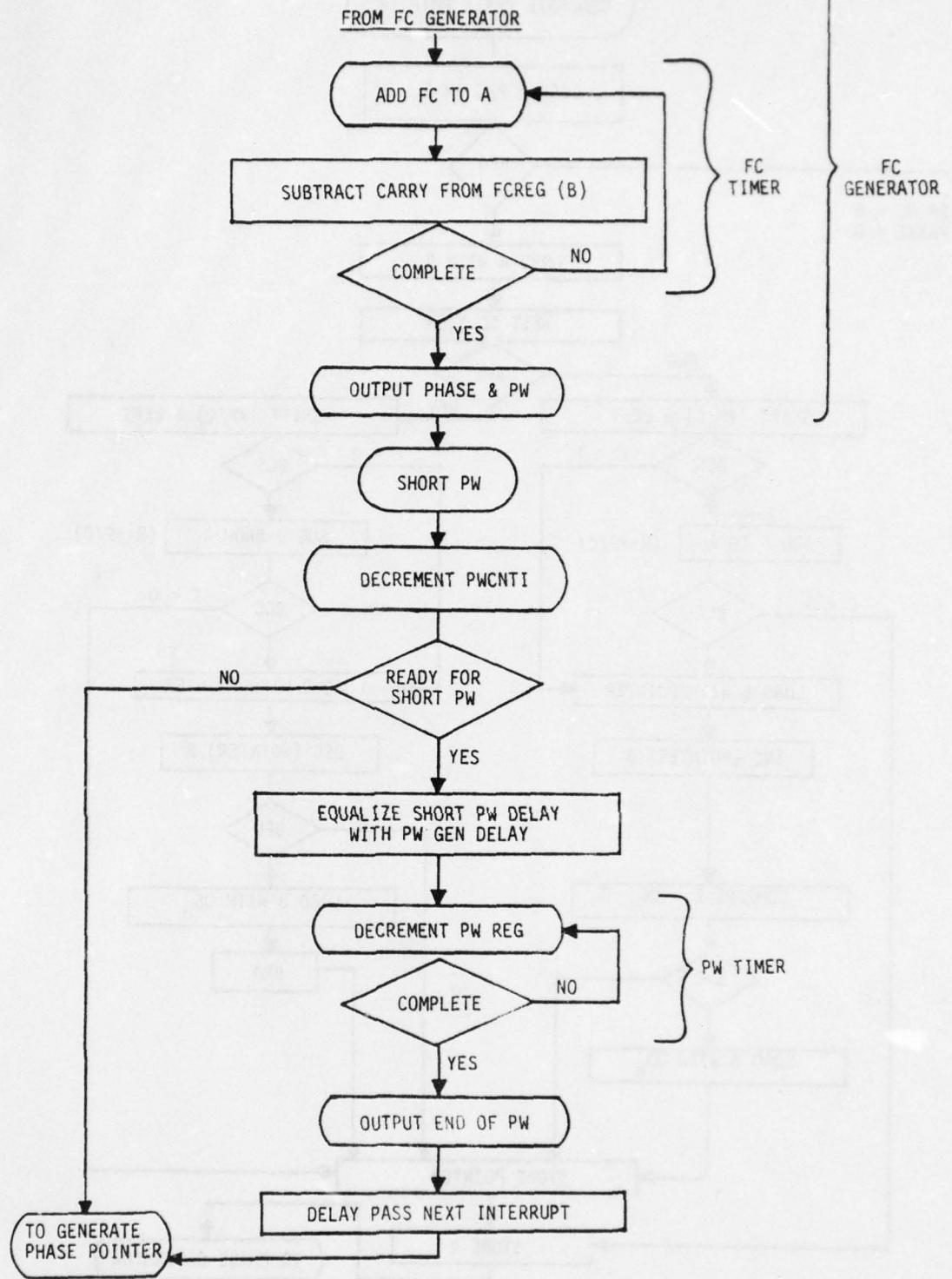


FIGURE 20. INNER LOOP 2.

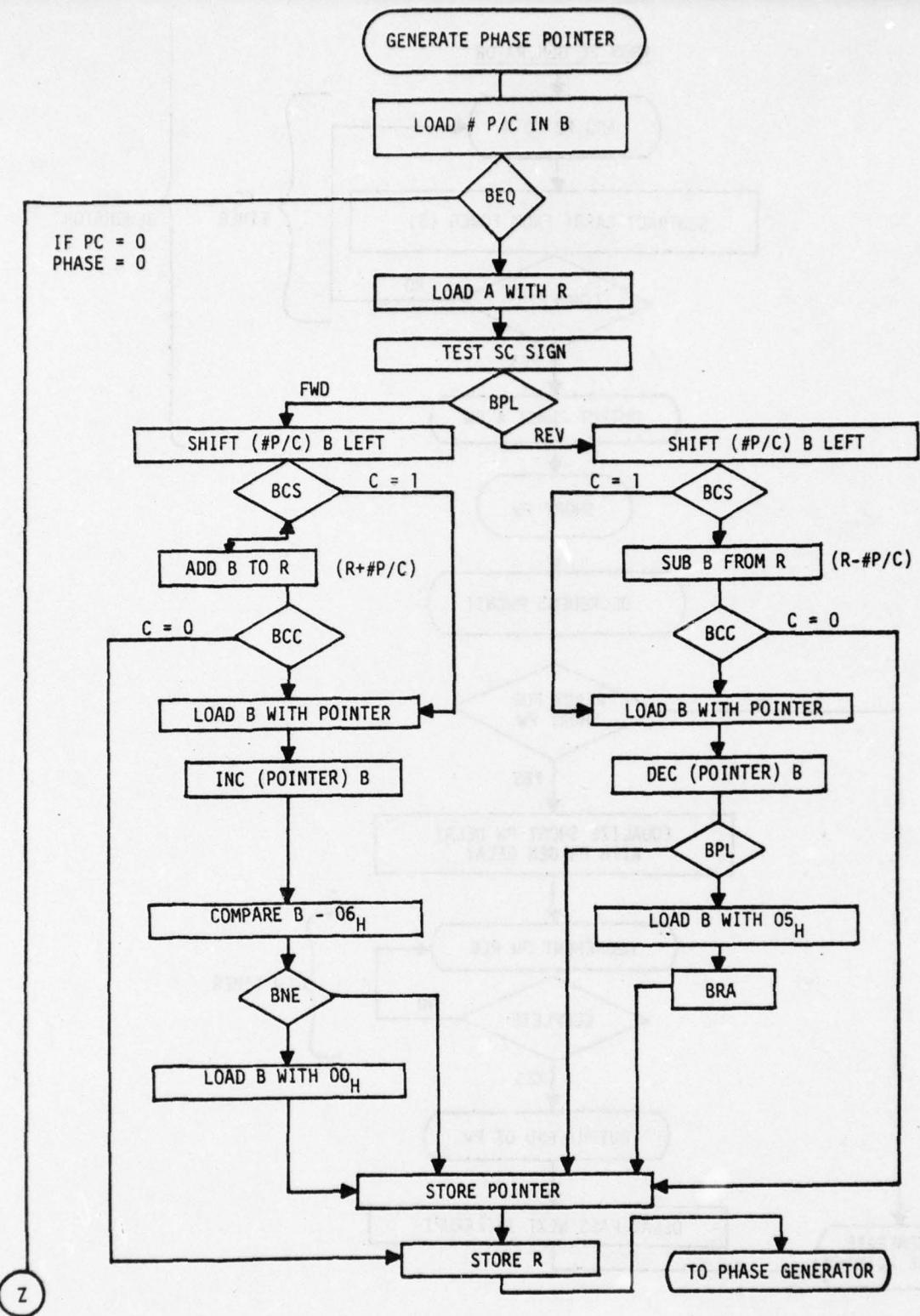


FIGURE 21. INNER LOOP 3.

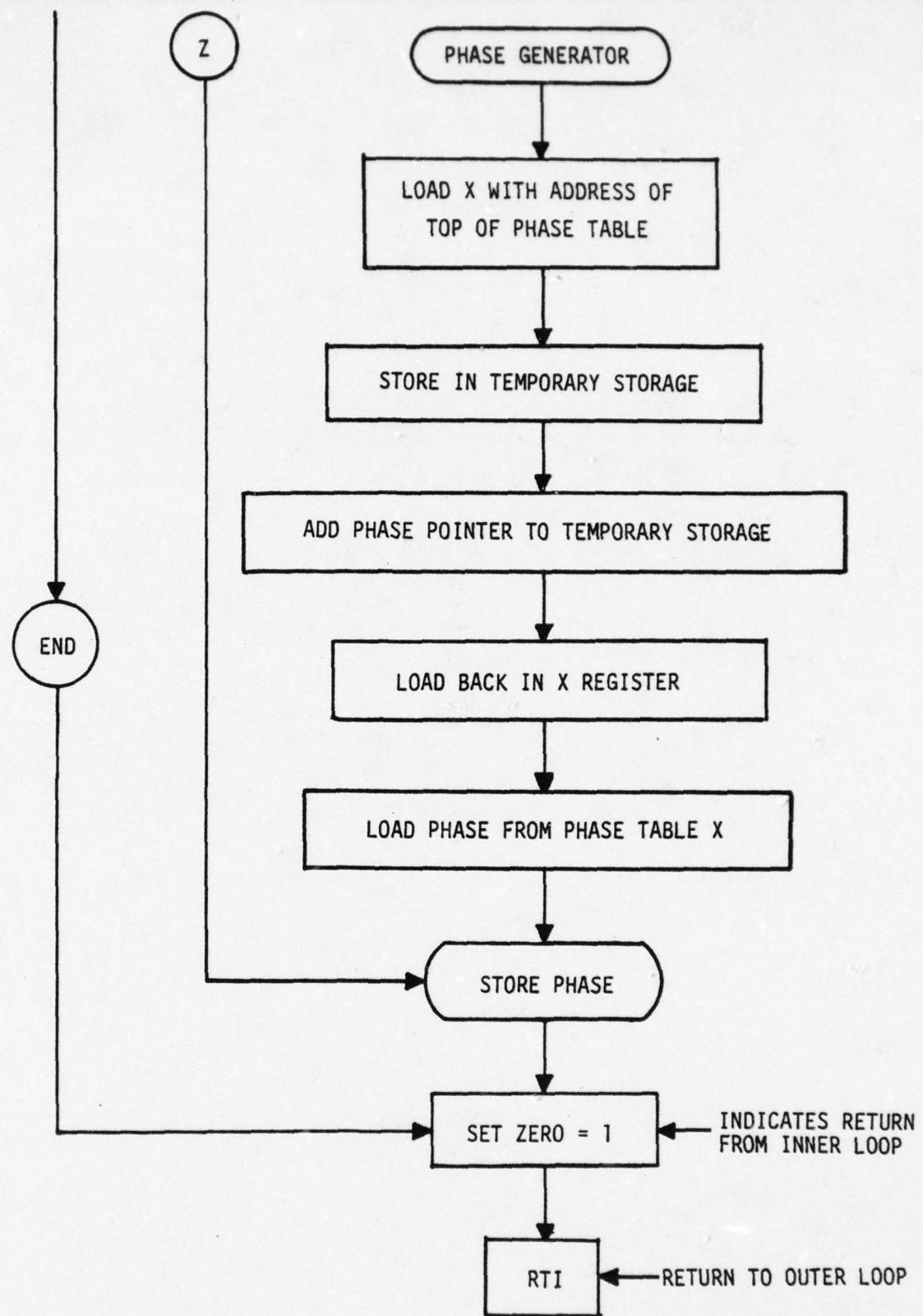


FIGURE 22. INNER LOOP 4.

APPENDIX A

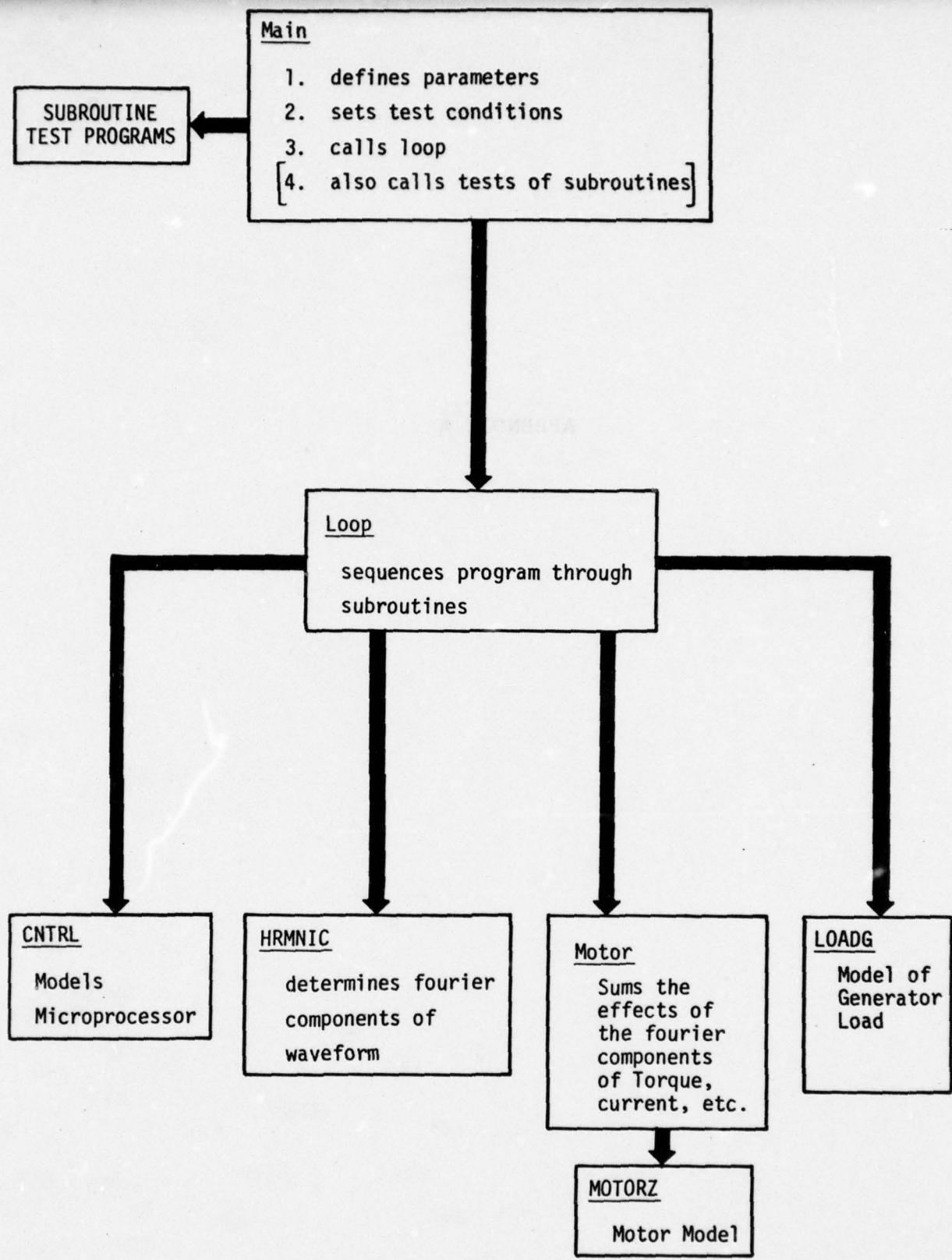


Figure 23. MODEL PROGRAM BLOCK DIAGRAM;
SIMULATION PROGRAM FLOW

LEVEL 2.2 (SEPT 76)

OS/360 FORTRAN H EXTENDED

DATE 78.030/12

REQUESTED OPTIONS: MAP,XREF,NAME=MAIN,OPT=0

OPTIONS IN EFFECT: NAME(MAIN) NOOPTIMIZE LINECOUNT(60) SIZE(MAX) AUTOUBL(NONE)
SOURCE EBCDIC NOLIST NODECK OBJECT MAP NOFORMAT GOSTMT XREF NOALC NOANSF NOTERM

C	TEST TEST TEST TEST TEST	00000010
C	SIMULATION --- FORTRAN	00000020
C	MICROPROSSER // MOTOR CONTROLLER	00000030
C		00000040
C	MC77SIM.FORT	00000050
C	TEST	00000060
C	TEST	00000070
C	TEST	00000080
C	INITIAL CONDITIONS	00000090
C		00000100
C		00000110
C		00000120
ISN 0002	INTEGER DZLO,DZHI,TLL1LO,TLL1HI,TLL2LO,TLL2HI,DLTADW,DLTAUP, 1 BNDRY1,BNDRY2,BNDRY4,BNDRY8,HYST,ILIMIT,IMAX,SIDEAL	00000130 00000140
C		00000150
ISN 0003	REAL R1,L1,RMC,LM,R2,L2	00000160
C		00000170
ISN 0004	REAL KG,J,STCTQ	00000180
C		00000190
ISN 0005	INTEGER SC,TAC,FC,PC,PW	00000200
ISN 0006	REAL I1,I2,VB,F,TORQ,PF,ROTORL,TIME,DT,V1,V5,V7,V11,V13, 1 WTAC,RDTAC,DT,WDLTUP,WDLTDW,WSC,WBNDY1,WBNDY2,WBNDY4	00000210 00000220
ISN 0007	LOGICAL FCSGN,INITC,SKIP,SKFOLD,SKSLIP	00000230
C		00000240
ISN 0008	COMMON /ONE/ DZLO,DZHI,TLL1LO,TLL1HI,TLL2LO,TLL2HI, 1 DLTADW,DLTAUP,BNDRY1,BNDRY2,BNDRY4,BNDRY8,HYST, 2 SKFOLD,SKSLIP,ILIMIT,IMAX,SIDEAL	00000250 00000260 00000270
C		00000280
ISN 0009	COMMON /TWO/ R1,L1,RMC,LM,R2,L2	00000290
C		00000300
ISN 0010	COMMON /THREE/ KG,J,STCTQ	00000310
C		00000320
ISN 0011	FOUR(A)=A*4.+01	00000330
ISN 0012	FOURN(B)=B*4.-01	00000340
C		00000350
C	FIXED DATA	00000360
ISN 0013	DZLO=FOUR(7.5)	00000370
ISN 0014	DZHI=FOUR(8)	00000380
ISN 0015	TLL1LO=FOUR(2.5)	00000390
ISN 0016	TLL1HI=FOUR(3)	00000400
ISN 0017	TLL2LO=FOUR(10)	00000410
ISN 0018	TLL2HI=FOUR(11)	00000420
ISN 0019	DLTADW=FOUR(7.5)	00000430
ISN 0020	DLTAUP=FOUR(7.5)	00000440
ISN 0021	BNDRY1=FOUR(45)	00000450
ISN 0022	BNDRY2=FOUR(30)	00000460
ISN 0023	BNDRY4=FOUR(10.)	00000470
ISN 0024	BNDRY8=FOUR(7.25)	00000480
ISN 0025	HYST=FOUR(1.)	00000490
ISN 0026	ILIMIT=FOUR(50.)	00000500
ISN 0027	IMAX=FOUR(50.*4./3.)	00000510
ISN 0028	SIDEAL=FOUR(5.)	00000520
C		00000530

LEVEL 2.2 (SEPT 76)

MAIN

OS/360 FORTRAN H EXTENDED

DATE 78.030/12.

```

C
C
C      MOTOR PARAMETERS  R OHMS, L HENRIES
C
C
C      ISN 0029      R1=.12          00000540
C      ISN 0030      L1=.00068       00000550
C      ISN 0031      RMC=.003        00000560
C      ISN 0032      LM=.013        00000570
C      ISN 0033      R2=.09         00000580
C      ISN 0034      L2=.00068       00000590
C
C
C      LOAD PARAMETERS
C
C      I1=45AMP @ 60HZ @ 5%SLIP @ V=82VOLTS
C      I2=40AMPS          00000600
C      RDTAC=2*PI*WTAC    00000610
C      TORQ=3*I2**2*R2/(RDTAC*SLIP) =20. NEWTON-METERS
C      TORQ=J*RDTAC' + KG*RDTAC +STCTQ (STICKTION TORQUE)
C      RDTAC=2*PI*F*(1-SLIP)
C      KG=(TQ-STCTQ)/RDTAC=21./377*.95 =.055 NEWTON-METERS-SEC**(-1)
C      ASSUME: STCTQ=.2 NEWTON METERS OR 1% OF RATED TORQUE
C      ASSUME: TIME CONSTANT=.3 SEC
C      TIMEC=J/KG
C      J=KG*TIMEC =.055*.3 =.0165 NEWTON-METERS-SEC**(-2)
C      MAX DT=.03SEC     USE DT=.01SEC
C
C      ISN 0035      KG=.055        00000800
C      ISN 0036      J=.0165        00000810
C      ISN 0037      STCTQ=.2      00000820
C
C
C      ISN 0038      INITC=.FALSE. 00000830
C      ISN 0039      SKFOLD=.FALSE. 00000840
C      ISN 0040      SKSLIP=.FALSE. 00000850
C
C
C      MAIN PROGRAM
C
C
C      TEST SUBROUTINES
C
C      ISN 0041      CALL TCNTRL   00000900
C      ISN 0042      CALL THRM      00000910
C      ISN 0043      CALL TLOADG    00000940
C      ISN 0044      CALL TMOTZ     00000950
C      ISN 0045      CALL TMOTOR    00000960
C
C
C      EXECUTE MC77SIM SIMULATION
C
C      ISN 0046      WRITE(6,1000)  00000970
C      ISN 0047      DO 101 K1=1,10   00000980
C      ISN 0048      WRITE(6,1001)  00001000
C      ISN 0049      101 CONTINUE   00001010
C
C
C      STEP FUNCTIONS
C
C

```

LEVEL 2.2 (SEPT 76)	MAIN	OS/360 FORTRAN H EXTENDED	DATE 78.030/12
ISN 0050	DO 100 K=1,10		00001140
C			00001150
ISN 0051	IF (K.EQ.1) SKIP=.FALSE.		00001160
ISN 0053	IF (K.EQ.2) SKIP=.TRUE.		00001170
ISN 0055	IF (K.EQ.3) DLT DUP=FOUR(3)		00001180
ISN 0057	IF (K.EQ.4) GO TO 100		00001190
ISN 0059	IF (K.EQ.5) DLT DUP=FOUR(15.)		00001200
ISN 0061	DLTADW=DLTAUP		00001210
C			00001220
ISN 0062	IF (K.LE.5)GO TO 120		00001230
C	TEST ADJUSTED CONDITIONS		00001240
ISN 0064	DLTAUP=FOUR(7.5)		00001250
ISN 0065	DLTADW=FOUR(3.)		00001260
ISN 0066	TLL2LO=FOUR(5.)		00001270
ISN 0067	TLL2HI=FOUR(5.5)		00001280
C			00001290
ISN 0068	IF (K.LE.6)GO TO 120		00001300
C	TEST HARMONIC EFFECTS		00001310
C	TEST 1P/C MODE		00001320
ISN 0070	BNDY4=FOUR(7.5)		00001330
ISN 0071	BNDY2=FOUR(7.75)		00001340
ISN 0072	BNDY1=FOUR(8.)		00001350
C	TEST 2P/C MODE		00001360
ISN 0073	IF (K.GE.8) BNDY1=FOUR(60.)		00001370
C	TEST 4P/C MODE		00001380
ISN 0075	IF (K.GE.9) BNDY2=FOUR(59.)		00001390
C	TEST 6P/C MODE		00001400
ISN 0077	IF (K.GE.10) BNDY4=FOUR(58.)		00001410
C			00001420
ISN 0079	120 CONTINUE		00001430
C			00001440
ISN 0080	WDLTUP=FLOAT(DLTAUP)/4.		00001450
ISN 0081	WDLTDW=FLOAT(DLTADW)/4.		00001460
ISN 0082	WBNDY1=FLOAT(BNDY1)/4.		00001470
ISN 0083	WBNDY2=FLOAT(BNDY2)/4.		00001480
ISN 0084	WBNDY4=FLOAT(BNDY4)/4.		00001490
C			00001500
ISN 0085	WRITE(6,1002) K		00001510
ISN 0086	WRITE(6,1050) SKIP,WDLTUP,WDLTDW,WBNDY1,WBNDY2,WBNDY4		00001520
ISN 0087	1000 FORMAT ('1',1X,26HEXECUTE MC77SIM SIMULATION)		00001530
ISN 0088	1001 FORMAT (1X,26HEXECUTE MC77SIM SIMULATION)		00001540
ISN 0089	1002 FORMAT ('1',1X,16HSTEP FUNCTION # ,I2)		00001550
ISN 0090	1050 FORMAT (1X,5HSKIP=,L1,2X,7HDLTAUP=,F5.2,2X,7HDLTADW=,F5.2,2X, 1 7HBNDY1=,F6.2,2X,7HBNDY2=,F6.2,2X,7HBNDY4=,F6.2)		00001560
C			00001570
ISN 0091	TIME=0.		00001580
ISN 0092	VS=105.		00001590
C	NOTE: RMS VALUE @ 100 %MOD = 82 VOLTS		00001600
ISN 0093	DT=.01		00001610
ISN 0094	WTAC=0.		00001620
ISN 0095	SC=FOUR(60.)		00001630
C			00001640
C	INITIALIZE CONTROL SUBROUTINE		00001650
ISN 0096	INITC=.TRUE.		00001660
ISN 0097	CALL LOOP(TIME,DT,VB,SC,WTAC,SKIP,INITC)		00001670
ISN 0098	INITC=.FALSE.		00001680
C			00001690
			00001700
			00001710

LEVEL 2.2 (SEPT 76)	MAIN	OS/360 FORTRAN H EXTENDED	DATE 78.030/12.
ISN 0099	DO 100 I=1,250		00001720
	C		00001730
ISN 0100	IF (I.GT.100) SC=FOURN(-60.)		00001740
ISN 0102	IF (I.GT.200) SC=FOUR(0.)		00001750
ISN 0104	IF (I.EQ.1)GO TO 110		00001760
ISN 0106	IF (I.EQ.101)GO TO 110		00001770
ISN 0108	IF (I.EQ.201)GO TO 110		00001780
	C		00001790
ISN 0110	130 CALL LOOP(TIME,DT,VB,SC,WTAC,SKIP,INITC)		00001800
	C		00001810
ISN 0111	GO TO 100		00001820
	C		00001830
ISN 0112	110 WSC=FLOAT(SC)/4.		00001840
	C		00001850
ISN 0113	WRITE(6,1010) WSC,WTAC,VB,TIME		00001860
ISN 0114	WRITE(6,1020)		00001870
ISN 0115	1010 FORMAT (//,IX,3HSC=,F6.2,2X,4HTAC=,F6.2,2X,3HV8=,F5.1,2X, 1 5HTIME=,F4.2)		00001880
ISN 0116	1020 FORMAT (/,1X,4X,4HTIME,3X,4HWHTAC,4X,4HTORQ,6X,2HI1,5X,3HIH%, 1 4X,4HV1NM,4X,4HVHNM,5X,3HVHZ,5X,3HEFF,6X,2HPF,5X, 2 3HI1H,4X,4HFREQ,6X,2HPW)		00001900
ISN 0117	GO TO 130		00001920
	C		00001930
ISN 0118	100 CONTINUE		00001940
	C		00001950
	C TEST SLIP CONTROL		00001960
	C		00001970
ISN 0119	210 SKIP=.TRUE.		00001980
ISN 0120	DLTAUP=FOUR(7.5)		00002000
ISN 0121	DLTDW=FOUR(3.)		00002010
ISN 0122	TLL2LO=FOUR(5.)		00002020
ISN 0123	TLL2HI=FOUR(5.5)		00002030
ISN 0124	BNDRY1=FOUR(45)		00002040
ISN 0125	BNDRY2=FOUR(30)		00002050
ISN 0126	BNDRY4=FOUR(10.)		00002060
ISN 0127	BNDRY8=FOUR(7.25)		00002070
	C		00002080
ISN 0128	DO 200 KSLIP=1,3		00002090
ISN 0129	TIME=0.		00002100
ISN 0130	DT=.01		00002110
ISN 0131	VB=105.		00002120
ISN 0132	SC=FOUR(60.)		00002130
ISN 0133	WTAC=57.		00002140
ISN 0134	KG=.055		00002150
ISN 0135	IF (KSLIP.EQ.2) SKSLIP=.TRUE.		00002160
ISN 0137	IF (KSLIP.EQ.3) SIDEAL=FOUR(3.)		00002165
	C		00002170
ISN 0139	IDEALS=SIDEAL/4		00002171
ISN 0140	WRITE(6,1030) SKSLIP,IDEALS		00002180
ISN 0141	1030 FORMAT ('1',1X,27HTEST SLIP CONTROL SKSLIP=,L1,9H SIDEAL=,I2)		00002190
ISN 0142	WRITE(6,1020)		00002200
	C INITIALIZE CONTROL SUBROUTINE		00002210
ISN 0143	INITC=.TRUE.		00002220
ISN 0144	CALL LOOP(TIME,DT,VB,SC,WTAC,SKIP,INITC)		00002230
ISN 0145	INITC=.FALSE.		00002240
	C		00002250
ISN 0146	DO 200 ISLIP=1,100		00002260
ISN 0147	SC=FOUR(60.)		00002265

LEVEL 2.2 (SEPT 76)	MAIN	OS/360 FORTRAN H EXTENDED	DATE 78.030/12
ISN 0148	CALL LOOP(TIME,DT,VB,SC,WTAC,SKIP,INITC)		00002270
ISN 0149	IF (ISLIP.LT.25) GO TO 200		00002280
ISN 0151	KG=KG-.001		00002290
ISN 0152	IF (KG.LT.0.0001) KG=.001		00002300
ISN 0154	200 CONTINUE		00002310
C			00002320
C	TEST SC I FOLDBACK		00002330
C			00002340
ISN 0155	DO 300 KFOLD=1,3		00002350
ISN 0156	TIME=0.		00002350
ISN 0157	DT=.01		00002370
ISN 0158	SKSLIP=.FALSE.		00002380
ISN 0159	SC=FOUR(60.)		00002390
ISN 0160	WTAC=-60.		00002400
ISN 0161	KG=.055		00002410
ISN 0162	IF (KFOLD.EQ.2) SKFOLD=.TRUE.		00002420
ISN 0164	IF (KFOLD.EQ.3) ILIMIT=FOUR(50.)		00002423
ISN 0166	IF (KFOLD.EQ.3) IMAX=FOUR(2.*50.)		00002426
C			00002430
ISN 0168	WRITE (6,1040) SKFOLD		00002440
ISN 0169	1040 FORMAT ('1',1X,28HTEST SC I FOLDBACK SKFOLD=,L1)		00002450
ISN 0170	WRITE (6,1020)		00002460
C	INITIALIZE CONTROL SUBROUTINE		00002470
ISN 0171	INITC=.TRUE.		00002480
ISN 0172	CALL LOOP(TIME,DT,VB,SC,WTAC,SKIP,INITC)		00002490
ISN 0173	INITC=.FALSE.		00002500
C			00002510
ISN 0174	DO 300 IFOLD=1,360		00002520
ISN 0175	IF (IFOLD.EQ.125) KG=.075		00002530
ISN 0177	IF (IFOLD.GE.300) KG=KG+.002		00002540
C			00002550
ISN 0179	SC=FOUR(60.)		00002555
ISN 0180	CALL LOOP(TIME,DT,VB,SC,WTAC,SKIP,INITC)		00002560
ISN 0181	300 CONTINUE		00002570
ISN 0182	STOP		00002580
ISN 0183	END		00002590

*****FORTRAN CROSS REFERENCE LISTING*****

SYMBOL	INTERNAL STATEMENT NUMBERS
F	0006
I	0099 0100 0102 0104 0106 0108
J	0004 0010 0036
K	0050 0051 0053 0055 0057 0059 0062 0068 0073 0075 0077 0085
DT	0006 0006 0093 0097 0110 0130 0144 0148 0157 0172 0180
FC	0005
I1	0006
I2	0006
KG	0004 0010 0035 0134 0151 0151 0152 0152 0161 0175 0177 0177
K1	0047
LM	0003 0009 0032
L1	0003 0009 0030
L2	0003 0009 0034
PC	0005
PF	0006
PW	0005
R1	0003 0009 0029
R2	0003 0009 0033

LEVEL 2.2 (SEPT 76)

OS/360 FORTRAN H EXTENDED

DATE 78.030/12

REQUESTED OPTIONS: MAP,XREF,NAME=LOOP,OPT=0

OPTIONS IN EFFECT: NAME(LOOP) NOOPTIMIZE LINECOUNT(60) SIZE(MAX) AUTODBL(NONE)
SOURCE EBCDIC NOLIST NODECK OBJECT MAP NOFORMAT GOSTMT XREF NOALC NOANSF NOTERM

	C	00000010
	C	00000020
ISN 0002	C	00000030
	C	00000040
ISN 0003	INTEGER SC,TAC,FC,PC,PW,I	00000050
ISN 0004	REAL II,I2,VB,F,TORQ,PF,ROTORL,TIME,DT,V1,V5,V7,V11,V13, 1 WTAC,STAC,RDTAC,DT,PI,W,FWOUT,PWIN,EFF,FREQ,VRMS, 2 V1NM,VHNM1,VHPCNT,II1,IIH,IHPCNT,FN,V17,V19,V23,V25	00000060 00000070 00000080
ISN 0005	LCGICAL FCSGN,INITC,SKIP	00000090
	C	00000100
ISN 0006	FOUR(A)=A*4.+.01	00000110
ISN 0007	FOURN(B)=B*4.-.01	00000120
	C	00000130
ISN 0008	IF (INITC) GO TO 105	00000140
ISN 0010	TIME=TIME+DT	00000150
	C	00000160
ISN 0011	IF (SKIP) GO TO 105	00000170
ISN 0013	FC=IABS(SC)	00000180
ISN 0014	FCSGN=.TRUE.	00000190
ISN 0015	IF (SC.LT.0) FCSGN=.FALSE.	00000200
ISN 0017	PC=1	00000210
ISN 0018	PW=2560	00000220
ISN 0019	GO TO 115	00000230
	C	00000240
ISN 0020	105 IF (WTAC.GE.0.) TAC=FOUR(WTAC)	00000250
ISN 0022	IF (WTAC.LT.0.) TAC=FOURN(WTAC)	00000260
	C	00000270
ISN 0024	I=FOUR(II)	00000275
ISN 0025	CALL CNTRL (SC,TAC,FC,FCSGN,PC,PW,INITC,I,DT)	00000280
	C	00000290
ISN 0026	CLEAR COMMAND	00000300
	115 IF (FC.GT.0)GO TO 110	00000310
ISN 0028	V1=0.	00000320
ISN 0029	V5=0.	00000330
ISN 0030	V7=0.	00000340
ISN 0031	V11=0.	00000350
ISN 0032	V13=0.	00000360
ISN 0033	V17=0.	00000370
ISN 0034	V19=0.	00000380
ISN 0035	V23=0.	00000390
ISN 0036	V25=0.	00000400
ISN 0037	F=0.	00000410
ISN 0038	GO TO 120	00000420
	C	00000430
ISN 0039	110 F=FLOAT(FC)/4.	00000440
	C	00000450
ISN 0040	CALL HRMNIC (F,PC,PW,VB,V1,V5,V7,V11,V13,V17,V19,V23,V25)	00000460
	C	00000470
ISN 0041	120 STAC=WTAC	00000480
ISN 0042	IF (.NOT.FCSGN)STAC=STAC*(-1.)	00000490
	C	00000500
ISN 0044	CALL MOTOR (V1,V5,V7,V11,V13,V17,V19,V23,V25, 1 F,STAC,II,I2,TORQ,ROTORL,PF,II1,IIH)	00000510 00000520

LEVEL 2.2 (SEPT 76)	LOOP	OS/360 FORTRAN H EXTENDED	DATE 78.030/12.
	C		00000530
ISN 0045	C IF (.NOT.FCSGN)TORQ=TORQ*(-1.)		00000540
	C PI=3.1416		00000550
ISN 0048	RD_TAC=2*PI*WTAC		00000570
	C CALL LOADG (TORQ, RD_TAC, DT)		00000580
ISN 0049	C		00000590
ISN 0050	WTAC=RD_TAC/(2*PI)		00000600
	C PWOUT=TORQ*RD_TAC		00000610
ISN 0051	VRMS=SQRT(V1**2+V5**2+V7**2+V11**2+V13**2		00000620
ISN 0052	1 +V17**2+V19**2+V23**2+V25**2)		00000630
ISN 0053	PWIN=3*I1*VRMS*PF		00000640
ISN 0054	EFF=0.		00000650
ISN 0055	IF (ABS(PWIN).GT.0.) EFF=(PWOUT/PWIN)*100.		00000660
	C FREQ=F		00000670
ISN 0057	IF (.NOT.FCSGN) FREQ=F*(-1.)		00000680
	C FN=F		00000690
ISN 0061	IF (FN.LT.0.01) FN=.01		00000700
ISN 0063	V1NM=V1*60./FN		00000710
ISN 0064	VHNM=(SQRT(V5**2+V7**2+V11**2+V13**2+		00000720
	1 V17**2+V19**2+V23**2+V25**2))*60./FN		00000730
ISN 0065	IF (V1NM.LT.0.1) V1NM=.1		00000740
ISN 0067	VHPCNT=(VHNM/V1NM)*100.		00000750
ISN 0068	IF (I1L.LT.0.1) I1L=.1		00000760
ISN 0070	IHPCNT=(I1H/I1L)*100.		00000770
	C WRITE(6,1001) TIME,WTAC,TORQ,I1,IHPCNT,V1NM,VHNM,VHPCNT,EFF,PF,		00000780
ISN 0071	1 I1H,FREQ,PW		00000790
ISN 0072	WRITE(14,1002) TIME,WTAC,TORQ,I1,IHPCNT,V1NM,VHNM,VHPCNT,EFF,PF,		00000800
	1 FREQ		00000810
ISN 0073	1001 FORMAT (1X,12F8.2,I8)		00000820
ISN 0074	1002 FORMAT (8F8.2,F4.0,F4.2,F4.0)		00000830
	C		00000840
ISN 0075	RETURN		00000850
ISN 0076	END		00000860

*****FORTRAN CROSS REFERENCE LISTING*****								
SYMBOL	INTERNAL STATEMENT NUMBERS							
F	0004	0037	0039	0040	0044	0057	0058	0060
I	0003	0024	0025					
W	0004							
DT	0002	0004	0004	0010	0025	0049		
FC	0003	0013	0025	0026	0039			
FN	0004	0060	0061	0061	0063	0064		
I1	0004	0024	0044	0053	0071	0072		
I2	0004	0044						
FC	0003	0017	0025	0040				
PF	0004	0044	0053	0071	0072			
PI	0004	0047	0048	0050				
FW	0003	0018	0025	0040	0071			
SC	0002	0003	0013	0015	0025			
VB	0002	0004	0040					
V1	0004	0028	0040	0044	0052	0063		

LEVEL 2.2 (SEPT 76)

OS/360 FORTRAN H EXTENDED

DATE 78.030/12

REQUESTED OPTIONS: MAP,XREF,NAME=CNTRL,OPT=0

OPTIONS IN EFFECT: NAME(CNTRL) NOOPTIMIZE LINECOUNT(60) SIZE(MAX) AUTODBL(NONE)
SOURCE EBCDIC NOLIST NODECK OBJECT MAP NOFORMAT GOSTMT XREF NOALC NOANSF NOTERM

	C	00000010
	C	00000020
	C	00000030
ISN 0002	SUBROUTINE CNTRL (SC,TAC,FC,FCSGN,PC,PW,INITC,I,DT)	00000040
	SET CONDITIONS	00000050
	C	00000060
	C	00000070
ISN 0003	INTEGER SC,TAC,DZ,TLL1,TLL2,BNDRY,DLTF,FC,PC,PW,SCTAC,	00000080
	1 I,ILIMIT,IMAX,SCFOLD,SIDEAL,SLTIME,DSCTAC,INCFLD	00000090
ISN 0004	LOGICAL SCSGN,TACSGN,DSGN,FCSGN,SLIPC,INITC,	00000100
	1 SKSLIP,SKFOLD	00000110
ISN 0005	REAL ITIME,DT,SCALE,SLIPN,SLIPSR,HYSTM	00000120
	C	00000130
	C	00000140
	C	00000150
	C	00000160
ISN 0006	INTEGER DZLO,DZHI,TLL1LO,TLL1HI,TLL2LO,TLL2HI,DLTADW,DLTAUP,	00000170
	1 BNDRY1,BNDRY2,BNDRY4,BNDRY8,HYST	00000180
ISN 0007	COMMON /ONEA/ DZLO,DZHI,TLL1LO,TLL1HI,TLL2LO,TLL2HI,	00000190
	1 DLTADW,DLTAUP,BNDRY1,BNDRY2,BNDRY4,BNDRY8,HYST,	00000200
	2 SKFOLD,SKSLIP,ILIMIT,IMAX,SIDEAL	00000210
	C	00000220
ISN 0008	FOUR(A)=A*4.+.01	00000230
	C	00000240
	C	00000250
ISN 0009	IF (.NOT.INITC)GO TO 185	00000260
ISN 0011	DZ=DZHI	00000270
ISN 0012	TLL1=TLL1HI	00000280
ISN 0013	TLL2=TLL2LO	00000290
ISN 0014	ITIME=0.	00000300
ISN 0015	HYSTM=1.	00000310
ISN 0016	INCFLD=0	00000320
ISN 0017	SCALE=0.	00000330
ISN 0018	SLTIME=0	00000340
ISN 0019	SLIPN=1.	00000350
ISN 0020	GO TO 500	00000360
	C	00000370
	C	00000380
	C	00000390
ISN 0021	185 MSC=IAES(SC)	00000400
ISN 0022	SCSGN=.TRUE.	00000410
ISN 0023	MTAC=IABS(TAC)	00000420
ISN 0024	TACSGN=.TRUE.	00000430
ISN 0025	IF (SC.LT.0) SCSGN=.FALSE.	00000440
ISN 0027	IF (TAC.LT.0) TACSGN=.FALSE.	00000450
	C	00000460
	C	00000470
	C	00000480
- ISN 0029	IF (.NOT.SKFOLD) GO TO 420	00000490
- ISN 0031	IF (INCFLD.GT.0) GO TO 440	00000500
ISN 0033	IF (I.GE.ILIMIT) GO TO 410	00000510
ISN 0035	ITIME=ITIME-DT	00000520
ISN 0036	IF (ITIME.GT.HYSTM) GO TO 450	00000530

LEVEL 2.2 (SEPT 76) CNTRL OS/360 FORTRAN H EXTENDED DATE 78.030/12.

```

ISN 0038      IF (ITIME.LT.0.) ITIME=0.          00000540
ISN 0040      HYSTM=1.                         00000550
ISN 0041      GO TO 420.                        00000560
ISN 0042      410 ITIME=ITIME+DT                00000570
ISN 0043      IF (ITIME.GE.2.0) ITIME=2.0       00000580
ISN 0045      IF (ITIME.LT.HYSTM) GO TO 420     00000590
ISN 0047      HYSTM=.5                         00000600
ISN 0048      INCFLD=5                         00000610
ISN 0049      450 SCALE=SCALE+((FLOAT(I-ILIMIT)/FLOAT(IMAX-ILIMIT))-SCALE)/8. 00000620
ISN 0050      440 SCFOLD=FOUR(60.*(1-SCALE))   00000630
ISN 0051      INCFLD=INCFLD-1                  00000640
ISN 0052      IF (INCFLD.LT.0) INCFLD=0        00000650
ISN 0054      IF (SCFOLD.LT.0) SCFOLD=0        00000660
ISN 0056      IF (SCFOLD.GT.FOUR(60.)) SCFOLD=FOUR(60.) 00000670
ISN 0058      IF (MSC.GT.SCFOLD) MSC=SCFOLD    00000680
ISN 0060      GO TO 430.                        00000690
ISN 0061      420 SCALE=SCALE/8.              00000700
ISN 0062      430 CONTINUE                      00000710
C
C
C      PERFORM SET CONDITIONS ROUTINE          00000720
C
ISN 0063      SLIPC=.FALSE.                    00000730
ISN 0064      FCSGN=SCSGN                   00000740
C
C
ISN 0065      IF (MSC.LT.DZ)GO TO 100          00000750
C
C      SC .GE. DZ                           00000760
ISN 0067      101 DZ=DZLO                     00000770
ISN 0068      IF (MTAC.LT.TLL1)GO TO 110     00000780
C
C      SC .GE. DZ & MOTOR RUNNING           00000790
ISN 0070      102 TLL1=TLL1LO                 00000800
ISN 0071      DSGN=(SCSGN.AND.TACSGN).OR.((.NOT.SCSGN).AND.(.NOT.TACSGN)) 00000810
ISN 0072      IF (.NOT.DSGN)GO TO 120        00000820
C
C      TACSGN=SCSGN                      00000830
ISN 0074      103 SCTAC=MSC-MTAC            00000840
ISN 0075      IF (SCTAC.LT.0)GO TO 130        00000850
C
C      SC .GE. TAC                          00000860
ISN 0077      104 IF (MSC.LE.(MTAC+DLTAUP))GO TO 170 00000870
C
C      MOTOR ACCELERATING                 00000880
ISN 0079      105 FC=MTAC+DLTAUP            00000890
ISN 0080      GO TO 190.                      00000900
C
C      TAC .GT. SC                         00000910
ISN 0081      130 IF (MSC.LT.(MTAC-DLTADW))GO TO 165 00000920
C
C      MOTOR SLOWING DOWN, TAC CLOSE TO SC 00000930
ISN 0083      135 FC=MSC                   00000940
ISN 0084      GO TO 190.                      00000950
C
C      SIGNS DIFFERENT                   00000960
ISN 0085      120 IF (MTAC.GE.TLL2)GO TO 140 00000970
C

```

LEVEL 2.2 (SEPT 76)	CNTRL	OS/360 FORTRAN H EXTENDED	DATE 78.030/12
	C	REVERSE MOTOR	00001120
ISN 0087	150	TLL2=TLL2HI	00001130
ISN 0088		FC=BNDRY8	00001140
ISN 0089		GO TO 190	00001150
	C		00001160
	C	MOTOR START COMMAND	00001170
ISN 0090	110	TLL1=TLL1HI	00001180
ISN 0091		FC=BNDRY8+MTAC	00001190
ISN 0092		GO TO 190	00001200
	C		00001210
	C	MOTOR STOPING OR REVERSING	00001220
ISN 0093	140	TLL2=TLL2LO	00001230
ISN 0094		FCSGN=TACSGN	00001240
ISN 0095		FC=MTAC-DLTADW	00001250
ISN 0096		SLIPC=.FALSE.	00001260
ISN 0097		GO TO 190	00001270
	C		00001280
	C	MOTOR SLOWING DOWN	00001290
ISN 0098	165	FC=MTAC-DLTADW	00001300
ISN 0099		SLIPC=.FALSE.	00001310
ISN 0100		FCSGN=SCSGN	00001320
ISN 0101		GO TO 190	00001330
	C		00001340
	C	SC .LT. DZ	00001350
ISN 0102	100	DZ=DZHI	00001360
ISN 0103		IF (MTAC.GE.TLL2) GO TO 140	00001370
	C		00001380
	C	ISSUE CLEAR COMMAND	00001390
ISN 0105		TLL2=TLL2HI	00001400
ISN 0106		GO TO 500	00001410
	C		00001420
	C	MOTOR CLOSE TO OR AT SPEED COMMAND	00001430
ISN 0107	170	FC=MSC	00001440
	C		00001450
	C	SLIPC=.FALSE. MEANS SKIP SLIP CONTROL ROUTINE	00001460
ISN 0108		SLIPC=.TRUE.	00001470
ISN 0109	190	CONTINUE	00001480
	C		00001490
	C		00001500
	C		00001510
	C	FC HYSTERESIS	00001520
ISN 0110		IF (PC.GT.8) GO TO 220	00001530
ISN 0112		GO TO(231,232,220,234,220,220,220,238),PC	00001540
ISN 0113	220	WRITE (6,2000)	00001550
ISN 0114	2000	FORMAT (1X,9H ERROR PC)	00001560
	C	NOTE: MULT NOT PRESENTLY USED	00001570
	C		00001580
ISN 0115	231	BNDRY=BNDRY1	00001590
ISN 0116		MULT=1	00001600
ISN 0117		GO TO 240	00001610
	C		00001620
ISN 0118	232	BNDRY=BNDRY2	00001630
ISN 0119		MULT=2	00001640
ISN 0120		GO TO 240	00001650
	C		00001660
ISN 0121	234	BNDRY=BNDRY4	00001670
ISN 0122		MULT=4	00001680
ISN 0123		GO TO 240	00001690

LEVEL 2.2 (SEPT 76) CNTRL OS/360 FORTRAN H EXTENDED DATE 78.030/12.

```

C
ISN 0124      238 BNDRY=BNDRY8          00001700
ISN 0125      MULT=8                  00001710
C
ISN 0126      240 DLT=FC-BNDRY          00001720
ISN 0127      IF (DLT.GE.0)GO TO 250    00001730
ISN 0129      IF (DLT.LT.(-HYST))GO TO 250 00001740
ISN 0131      FC=BNDRY              00001750
ISN 0132      250 CONTINUE            00001760
C
C
ISN 0133      DETERMINE #P/C &MAX PW 00001770
IF (FC.GE.BNDRY1)GO TO 310
ISN 0135      IF (FC.GE.BNDRY2)GO TO 320 00001780
ISN 0137      IF (FC.GE.BNDRY4)GO TO 330 00001790
C
C
ISN 0139      8 PULSE MODE           00001800
PC=8
ISN 0140      PW=320                 00001810
ISN 0141      GO TO 350               00001820
C
C
ISN 0142      1 PULSE MODE           00001830
310 PC=1
ISN 0143      PW=2560                00001840
ISN 0144      GO TO 350               00001850
C
C
ISN 0145      2 PULSE MODE           00001860
320 PC=2
ISN 0146      PW=1280                00001870
ISN 0147      GO TO 350               00001880
C
C
ISN 0148      4 PULSE MODE           00001890
330 PC=4
ISN 0149      PW=640                 00001900
ISN 0150      350 CONTINUE            00001910
C
C
ISN 0151      CORRECT FOR DLT          00001920
IF (DLT.GE.0)GO TO 370
ISN 0153      IF (DLT.LT.(-HYST))GO TO 370 00001930
ISN 0155      FC=FC+DLT              00001940
ISN 0156      370 CONTINUE            00001950
C
C
ISN 0157      SLIP CONTROL           00001960
IF (.NOT.SLIP) GO TO 610
ISN 0159      IF (.NOT.SLIPC) GO TO 610 00001970
ISN 0161      IF (SCTAC.LT.SIDEAL) GO TO 620 00001980
ISN 0163      SLIPN=SLIPN+(1-SLIPN)/2. 00001990
ISN 0164      GO TO 630               00002000
ISN 0165      610 SLIPN=1.             00002010
ISN 0166      GO TO 510               00002020
ISN 0167      620 SLTIME=SLTIME+1     00002030
C
ISN 0168      IF (SLTIME.LE.3) GO TO 630 00002040
SLTIME=0
ISN 0169      SLIPN=SLIPN+((FLOAT(SCTAC)/FLOAT(SIDEAL))-SLIPN)/4. 00002050
ISN 0170      630 CONTINUE            00002060
ISN 0171      IF (SLIPN.GT.1.0) SLIPN=1.0 00002070
ISN 0173      IF (SLIPN.LT.0.0) SLIPN=0.0 00002080
  
```

LEVEL 2.2 (SEPT 76)

CNTRL

OS/360 FORTRAN H EXTENDED

DATE 76.030/12

```

      C      CALCULATE MODIFIED SQUARE ROOT          00002280
  ISN 0175      IF (SLIPN.GE.0.5) SLIPSR=(SLIPN/2.)+.5  00002290
  ISN 0177      IF (SLIPN.LT.0.5) SLIFSR=SLIPN+.25   00002300
  ISN 0179      IF (SLIPN.LT.0.25) SLIPSR=SLIPN*.25  00002310
  ISN 0181      PW=PW*SLIPSR                      00002320
  ISN 0182      GO TO 510                         00002350
      C
      C      CLEAR CONDITIONS                     00002360
  ISN 0183      500 FC=0                          00002380
  ISN 0184      PC=8                           00002390
  ISN 0185      PW=320                         00002400
  ISN 0186      FCSSGN=.TRUE.                   00002410
  ISN 0187      SLIPN=1                        00002420
      C
  ISN 0188      510 RETURN                      00002430
  ISN 0189      END                           00002440
                                         00002450

```

*****FORTRAN CROSS REFERENCE LISTING*****

SYMBOL	INTERNAL STATEMENT NUMBERS
I	0002 0003 0033 0049
DT	0002 0005 0035 0042
DZ	0003 0011 0065 0067 0102
FC	0002 0003 0079 0083 0088 0091 0095 0098 0107 0126 0131 0133 0135 0137 0155
PC	0002 0003 0110 0112 0139 0142 0145 0148 0184
PW	0002 0003 0140 0143 0146 0149 0181 0185
SC	0002 0003 0021 0025
MSC	0021 0058 0058 0065 0074 0077 0081 0083 0107
TAC	0002 0003 0023 0027
DLTF	0003 0126 0127 0129 0151 0153 0155
DSGN	0004 0071 0072
DZHI	0006 0007 0011 0102
DZLO	0006 0007 0067
FOUR	0008 0050 0056 0056
HYST	0006 0007 0129 0153
IABS	0021 0023
IMAX	0003 0007 0049
MTAC	0023 0068 0074 0077 0079 0081 0085 0091 0095 0098 0103
MULT	0116 0119 0122 0125
TLL1	0003 0012 0068 0070 0090
TLL2	0003 0013 0085 0087 0093 0103 0105
BNDRY	0003 0115 0118 0121 0124 0126 0131
CNTRL	0002
FCSSN	0002 0004 0064 0094 0100 0186
FLOAT	0049 0049 0169 0169
HYSTM	0005 0015 0036 0040 0045 0047
INITC	0002 0004 0009
ITIME	0005 0014 0035 0035 0036 0038 0038 0042 0042 0043 0043 0045
SCALE	0005 0017 0049 0049 0049 0050 0061 0061
SCSGN	0004 0022 0025 0064 0071 0071 0100
SCTAC	0003 0074 0075 0161 0169
SLIPC	0004 0063 0096 0099 0103 0159
SLIPN	0005 0019 0163 0163 0163 0165 0169 0169 0169 0171 0171 0173 0173 0175 0175
	0187
BNDRY1	0006 0007 0115 0133
BNDRY2	0006 0007 0118 0135
BNDRY4	0006 0007 0121 0137
BNDRY8	0006 0007 0068 0091 0124

LEVEL 2.2 (SEPT 76)

OS/360 FORTRAN H EXTENDED

DATE 78.030/12

REQUESTED OPTIONS: MAP,XREF,NAME=HRMNIC,OPT=0

OPTIONS IN EFFECT: NAME(HRMNIC) NOOPTIMIZE LINECOUNT(60) SIZE(MAX) AUTOBL(NONE)
SOURCE EBCDIC NOLIST NODECK OBJECT MAP NOFORMAT GOSTMT XREF NOALC NOANSF NOTERM

C		00000010
C		00000020
C		00000030
ISN 0002	SUBROUTINE HRMNIC (F,PC,PW,VB,V1,V5,V7,V11,V13,V17,V19,V23,V25)	00000040
ISN 0003	INTEGER PC,PW	00000050
ISN 0004	REAL V1,V5,V7,V11,V13,V17,V19,V23,V25,F,DTR,DLTA,VK,VB,SUMA,	00000060
	1 PERIOD,PCMOD	00000070
C		00000080
ISN 0005	PERIOD=1000000./(6.*F)	00000090
ISN 0006	PCMOD=(PC*PW)/PERIOD	00000100
ISN 0007	DTR=3.1416/180.	00000110
ISN 0008	DLT A=60.* (1.-FCMOD)/(2.*PC)	00000120
ISN 0009	VK=4.*VB/(3.1416*1.4142)	00000130
C	CALCULATE FOR EACH HARMONIC	00000140
ISN 0010	DO 110 N=1,25,2	00000150
ISN 0011	IF (N.EQ.3)GO TO 110	00000160
ISN 0013	IF (N.EQ.9)GO TO 110	00000170
ISN 0015	IF (N.EQ.15)GO TO 110	00000180
ISN 0017	IF (N.EQ.21)GO TO 110	00000190
ISN 0019	SUMA=0.	00000200
ISN 0020	DO 120 J=1,PC,1	00000210
ISN 0021	ANGLEP=(30+(J-1)*60/PC+DLTA)*N	00000220
ISN 0022	ANGLEM=(30+J*60/PC-DLT A)*N	00000230
ISN 0023	SUMA=COS(ANGLEP*DTR)-COS(ANGLEM*DTR)+SUMA	00000240
ISN 0024	120 CONTINUE	00000250
ISN 0025	GO TO (131,110,110,110,132,110,133,110,110,134,110,135, 1 110,110,110,136,110,137,110,110,110,138,110,139),N	00000260 00000270
ISN 0026	131 V1=VK*SUMA/N	00000280
ISN 0027	GO TO 110	00000290
ISN 0028	132 V5=VK*SUMA/N	00000300
ISN 0029	GO TO 110	00000310
ISN 0030	133 V7=VK*SUMA/N	00000320
ISN 0031	GO TO 110	00000330
ISN 0032	134 V11=VK*SUMA/N	00000340
ISN 0033	GO TO 110	00000350
ISN 0034	135 V13=VK*SUMA/N	00000360
ISN 0035	GO TO 110	00000370
ISN 0036	136 V17=VK*SUMA/N	00000380
ISN 0037	GO TO 110	00000390
ISN 0038	137 V19=VK*SUMA/N	00000400
ISN 0039	GO TO 110	00000410
ISN 0040	138 V23=VK*SUMA/N	00000420
ISN 0041	GO TO 110	00000430
ISN 0042	139 V25=VK*SUMA/N	00000440
ISN 0043	110 CONTINUE	00000450
ISN 0044	RETURN	00000460
ISN 0045	END	00000470

*****FORTRAN CROSS REFERENCE LISTING*****
 SYMBOL INTERNAL STATEMENT NUMBERS
 F 0002 0004 0005
 J 0020 0021 0022

LEVEL 2.2 (SEPT 76)

OS/360 FORTRAN H EXTENDED

DATE 78.030/12

REQUESTED OPTIONS: MAP,XREF,NAME=MOTOR,OPT=0

OPTIONS IN EFFECT: NAME(MOTOR) NOOPTIMIZE LINECOUNT(60) SIZE(MAX) AUTODBL(NONE)
SOURCE EBCDIC NOLIST NCDECK OBJECT MAP NOFORMAT GOSTMT XREF NOALC NOANSF NOTERM

	C	00000010
	C	00000020
ISN 0002	SUBROUTINE MOTOR (V1,V5,V7,V11,V13,V17,V19,V23,V25, 1 F,STAC,I1,I2,TORQ,ROTORL,PF,I11,I1H)	00000030
	C	00000040
	C C CALCULATE MOTOR CURRENT I1, TORQUE T, POWER FACTOR PF, C C ROTOR LOSS ROTORL	00000050
	C	00000060
ISN 0003	REAL F,STAC,V1,V5,V7,V11,V13,I1,I11,I15,I17,I111,I113, 1 V17,V19,V23,V25,I117,I119,I123,I125, 2 I2,I21,I25,I27,I211,I213,TORQ,T1,T5,T7,T11,T13, 3 I217,I219,I223,I225,T17,T19,T23,T25, 4 ROTORL,ROTL1,ROTL5,ROTL7,ROTL11,ROTL13, 5 ROTL17,ROTL19,ROTL23,ROTL25, 6 PF,PF1,PF5,PF7,PF11,PF13,I1H, 7 PF17,PF19,PF23,PF25	00000070
	C	00000080
	C C 1ST HARMONIC	00000090
ISN 0004	N=1	00000100
ISN 0005	CALL MOTORZ(V1,N,F,I11,I21,STAC,T1,ROTL1,PF1)	00000110
	C 5TH HARMONIC	00000120
ISN 0006	N=5	00000130
ISN 0007	CALL MOTORZ(V5,N,F,I15,I25,STAC,T5,ROTL5,PF5)	00000140
	C 7TH HARMONIC	00000150
ISN 0008	N=7	00000160
ISN 0009	CALL MOTORZ(V7,N,F,I17,I27,STAC,T7,ROTL7,PF7)	00000170
	C 11TH HARMONIC	00000180
ISN 0010	N=11	00000190
ISN 0011	CALL MOTORZ(V11,N,F,I111,I211,STAC,T11,ROTL11,PF11)	00000200
	C 13TH HARMONIC	00000210
ISN 0012	N=13	00000220
ISN 0013	CALL MOTORZ(V13,N,F,I113,I213,STAC,T13,ROTL13,PF13)	00000230
	C 17TH HARMONIC	00000240
ISN 0014	N=17	00000250
ISN 0015	CALL MOTORZ(V17,N,F,I117,I217,STAC,T17,ROTL17,PF17)	00000260
	C 19TH HARMONIC	00000270
ISN 0016	N=19	00000280
ISN 0017	CALL MOTORZ(V19,N,F,I119,I219,STAC,T19,ROTL19,PF19)	00000290
	C 23TH HARMONIC	00000300
ISN 0018	N=23	00000310
ISN 0019	CALL MOTORZ(V23,N,F,I123,I223,STAC,T23,ROTL23,PF23)	00000320
	C 25TH HARMONIC	00000330
ISN 0020	N=25	00000340
ISN 0021	CALL MOTORZ(V25,N,F,I125,I225,STAC,T25,ROTL25,PF25)	00000350
	C	00000360
ISN 0022	I1=SQRT((I11**2)+(I15**2)+(I17**2)+(I111**2)+(I113**2) 1 +(I117**2)+(I119**2)+(I123**2)+(I125**2))	00000370
	I2=SQRT((I21**2)+(I25**2)+(I27**2)+(I211**2)+(I213**2) 1 +(I217**2)+(I219**2)+(I223**2)+(I225**2))	00000380
ISN 0024	I1H=SQRT((I15**2)*(I17**2)*(I111**2)*(I113**2) 1 +(I117**2)*(I119**2)*(I123**2)*(I125**2))	00000390
ISN 0025	TORQ=T1-T5+T7-T11+T13-T17+T19-T23+T25	00000400
ISN 0026	ROTORL=ROTL1+ROTL5+ROTL7+ROTL11+ROTL13+ROTL17+ROTL19+ROTL23+ROTL25 00000410	00000420
		00000430
		00000440
		00000450
		00000460
		00000470
		00000480
		00000490
		00000500
		00000510
		00000520
		00000530

LEVEL 2.2 (SEPT 76)

MOTOR

05/360 FORTRAN H EXTENDED

DATE 78.030/12.

ISN 0027	PF=PF1	00000540
ISN 0028	RETURN	00000550
ISN 0029	END	00000560

*****FORTRAN CROSS REFERENCE LISTING*****
SYMBOL INTERNAL STATEMENT NUMBERS
F 0002 0003 0005 0007 0009 0011 0013 0015 0017 0019 0021
N 0004 0005 0006 0007 0008 0009 0010 0011 0012 0013 0014 0015 0016 0017 0018 0
I1 0002 0003 0022
I2 0002 0003 0023
FF 0002 0003 0027
T1 0003 0005 0025
T5 0003 0007 0025
T7 0003 0009 0025
V1 0002 0003 0005
V5 0002 0003 0007
V7 0002 0003 0009
I1H 0002 0003 0024
I11 0002 0003 0005 0022
I15 0003 0007 0022 0024
I17 0003 0009 0022 0024
I21 0003 0005 0023
I25 0003 0007 0023
I27 0003 0009 0023
PF1 0003 0005 0027
PF5 0003 0007
FF7 0003 0009
T11 0003 0011 0025
T13 0003 0013 0025
T17 0003 0015 0025
T19 0003 0017 0025
T23 0003 0019 0025
T25 0003 0021 0025
V11 0002 0003 0011
V13 0002 0003 0013
V17 0002 0003 0015
V19 0002 0003 0017
V23 0002 0003 0019
V25 0002 0003 0021
I111 0003 0011 0022 0024
I113 0003 0013 0022 0024
I117 0003 0015 0022 0024
I119 0003 0017 0022 0024
I123 0003 0019 0022 0024
I125 0003 0021 0022 0024
I211 0003 0011 0023
I213 0003 0013 0023
I217 0003 0015 0023
I219 0003 0017 0023
I223 0003 0019 0023
I225 0003 0021 0023
PF11 0003 0011
PF13 0003 0013
PF17 0003 0015
PF19 0003 0017
PF23 0003 0019
PF25 0003 0021

LEVEL 2.2 (SEPT 76)

OS/360 FORTRAN H EXTENDED

DATE 78.030/12

REQUESTED OPTIONS: MAP,XREF,NAME=MOTORZ,OPT=0

OPTIONS IN EFFECT: NAME(MOTORZ) NOOPTIMIZE LINECOUNT(60) SIZE(MAX) AUTODBL(NONE)
SOURCE EBCDIC NOLIST NODECK OBJECT MAP NOFORMAT GOSTMT XREF NOALC NOANSF NOTERM

```

C          00000010
C          00000020
C          00000030
C          00000040
C          00000050
C          00000060
C          00000070
C          00000080
C          00000090
C          00000100
C          00000110
C          00000120
C          00000130
C          00000140
C          00000150
C          00000160
C          00000170
C          00000180
C          00000190
C          00000200
C          00000210
C          00000220
C          00000230
C          00000240
C          00000250
C          00000260
C          00000270
C          00000280
C          00000290
C          00000300
C          00000310
C          00000320
C          00000330
C          00000340
C          00000350
C          00000360
C          00000370
C          00000380
C          00000390
C          00000400
C          00000410
C          00000420
C          00000430
C          00000440
C          00000450
C          00000460
C          00000465
C          00000470
C          00000480
C          00000490
C          00000500
C          00000510
C          00000520

ISM 0002      SUBROUTINE MOTORZ(V,N,F,I1,I2,STAC,TQ,ROTL,PF)          00000040
C          CALCULATE MOTOR IMPEDANCE, CURRENT, POWER FACTOR,          00000050
C          AND ROTOR LOSS AT EACH HARMONIC          00000060
C          -----R1---L1-----+-----L2-----+          00000070
C          ----->           '           ----->           '
C          I1            RM           I2           '
C          '           R2/S<----+          00000110
C          '           '           '           '
C          LM           '           '           '
C          '           '           '           '
C          -----+-----+-----+          00000150
C          '
C          '
ISM 0003      REAL R1,L1,RMC,LM,R2,L2          00000190
ISM 0004      COMMON /TWO/ R1,L1,RMC,LM,R2,L2          00000200
ISM 0005      REAL RM,PI,W,ZPNR,ZPNX,ZPDR,ZPDX,ZPD,RP,XP,RI,XI,ZI,Z2,
1      TQ,SLIP,V,VP,F,ROTL,PF,I1,I2,ZP,STAC          00000210
C          ZP=ZM*Z2/(ZM+Z2)          00000220
C          '
C          (RM+JWL*LM)*(R2/S+JWL*L2)          00000230
C          ZP=-----          00000240
C          (RM+R2/S)+JW(LM+L2)          00000250
C          '
C          (RM*R2/S-(LM*L2)*W**2)+JW(RM*L2+LM*R2/S)          00000260
C          ZP=-----          00000270
C          (      ) + (      )          00000280
C          '
C          ( ZPNR + J ZPNX )      ( ZPDR - J ZPDX )          00000290
C          ZP=----- * -----          00000300
C          ( ZPDR + J ZPDX )      ( ZPDR - J ZPDX )          00000310
C          '
C          ZPD=(ZPDR)**2 + (ZPDX)**2          00000320
C          '
C          ZP = RP + J XP          00000330
C          '
ISM 0006      IF (ABS(F).LT..01) F=.01          00000340
C          '
ISM 0008      12 RM=RMC*F**1.25          00000350
C          '
ISM 0009      PI=3.1416          00000360
ISM 0010      W=2*PI*N*F          00000370
C          '

```

LEVEL 2.2 (SEPT 76) MOTORZ OS/360 FORTRAN H EXTENDED DATE 78.030/12.
 ISN 0011 SLIP=(N*F-STAC)/(N*F) 00000530
 ISN 0012 IF ((SLIP.LT.0.0001).AND.(SLIP.GT.(-0.0001))) SLIP=0.0001 00000540
 C
 ISN 0014 13 ZPNR=RM*R2/SLIP-(LM*L2)*W**2 00000550
 ISN 0015 14 ZPNX=W*(RM*L2+LM*R2/SLIP) 00000560
 ISN 0016 15 ZPDR=RM*R2/SLIP 00000570
 ISN 0017 16 ZFDX=W*(LM+L2) 00000580
 ISN 0018 17 ZPD=ZPDR**2+ZFDX**2 00000590
 ISN 0019 18 RP=(ZPNR*ZPDR+ZPNX*ZFDX)/ZPD 00000600
 ISN 0020 19 XP=(ZPNX*ZPDR-ZPNR*ZFDX)/ZPD 00000610
 ISN 0021 20 ZP=SQRT(RP**2+XP**2) 00000620
 C
 C ZI=Z1+ZP =(R1+RP) + (X1+XP) 00000630
 C = RI + J XI 00000640
 C
 ISN 0022 21 RI=R1+RP 00000650
 ISN 0023 22 XI=W*L1+XP 00000660
 ISN 0024 23 ZI=SQRT(RI**2+XI**2) 00000670
 C
 ISN 0025 24 I1=V/ZI 00000680
 C
 ISN 0026 25 VP=I1*ZP 00000690
 C
 ISN 0027 26 Z2=SQRT((R2/SLIP)**2+(W*L2)**2) 00000700
 C
 ISN 0028 27 I2=VP/Z2 00000710
 C
 C
 C TORQUE IN NEWTON-METERS 00000720
 ISN 0029 28 TQ=3*(I2**2)*R2/(W*SLIP) 00000730
 C
 ISN 0030 29 ROTL=3*R2*I2**2 00000740
 C
 C POWER FACTOR PF 1ST HARMONIC ONLY < REMOVE C 00000750
 C IF (N.GT.1)GO TO 300 00000760
 ISN 0031 30 PF=RI/ZI 00000770
 C
 C 300 CONTINUE 00000780
 C 31 WRITE(6,3336) RM,ZPNR,ZPNX,ZPDR,ZFDX,ZPD,RP,XP,ZP,RI,XI,ZI,Z2,I2 00000790
 C 3336 FORMAT (/,1X,F9.3,F9.3,F9.3,F9.3,F9.3,F9.1,F9.3,F9.3,F9.3,
 C 1 F9.3,F9.3,F9.3,F9.3,F8.1) 00000800
 C
 ISN 0032 RETURN 00000810
 ISN 0033 END 00000820
 00000830
 00000840
 00000850
 00000860
 00000870
 00000880
 00000890
 00000900
 00000910
 00000920
 00000930
 00000940
 00000950
 00000960

*****FORTRAN CROSS REFERENCE LISTING*****
 SYMBOL INTERNAL STATEMENT NUMBERS
 F 0002 0005 0006 0006 0008 0010 0011 0011
 N 0002 0010 0011 0011
 V 0002 0005 0025
 W 0005 0010 0014 0015 0017 0023 0027 0029
 I1 0002 0005 0025 0026
 I2 0002 0005 0028 0029 0030
 LM 0003 0004 0014 0015 0017
 L1 0003 0004 0023
 L2 0003 0004 0014 0015 0017 0027
 PF 0002 0005 0031

LEVEL 2.2 (SEPT 76) OS/360 FORTRAN H EXTENDED DATE 78.030/12.07.57 PAGE 1

REQUESTED OPTIONS: MAP,XREF,NAME=LOADG,OPT=0

OPTIONS IN EFFECT: NAME(LOADG) NOOPTIMIZE LINECOUNT(60) SIZE(MAX) AUTOUBL(NONE)
SOURCE EBCDIC NOLIST NCDECK OBJECT MAP NOFORMAT GOSTMT XREF NOALC NOANSF NOTERM FLAG(I)

```

ISN 0002      SUBROUTINE LOADG(TCR3,ROTAC,DT)
C          PPOWER=VW*2/R =((KV)**2/R)*ROTAC**2 =KG*ROTAC**2 =T*ROTAC
C          T=(JL+JM)*ROTAC'+KG*ROTAC*STCTQ =J*ROTAC'+KG*ROTAC*STCTQ
C          (ROTAC(T+DT)-ROTAC(T))/DT=-(KG/J)*ROTAC(T)+(T-STCTQ)/J
C          ROTAC(T+DT)=(1-DT*KG/J)*ROTAC(T)+(T-STCTQ)*DT/J
ISN 0003      REAL KG,J,DT,TORQ,ROTAC,STCTQ
ISN 0004      COMMON /THREE/KG,J,STCTQ
ISN 0005      IF(ROTAC.LT.0.) STCTQ=STCTQ*(-1)
ISN 0007      ROTAC=(1-DT*KG/J)*ROTAC+(TORQ-STCTQ)*DT/J
ISN 0008      RETURN
ISN 0009      END

```

*****F O R T R A N C R O S S R E F E R E N C E L I S T I N G *****

SYMBOL	INTERNAL STATEMENT NUMBERS
J	0003 0004 0007 0007
DT	0002 0003 0007 0007
KG	0003 0004 0007
TORQ	0002 0003 0007
LOADG	0002
ROTAC	0022 0003 0005 0007 0007
STCTQ	0003 0004 0005 0005 0007

/ LOADG / SIZE OF PROGRAM 00017A HEXADECIMAL BYTES

NAME	TAG	TYPE	ADD.	NAME	TAG	TYPE	ADD.	NAME	TAG	TYPE	ADD.	NAME	TAG	TYPE	ADD.		
J	F	C	R*4	000004	DT	F	R*4	000094	KG	F	C	R*4	000000	TORQ	F	R*4	00009E
LOADG			I*4	00009C	ROTAC	SF	R*4	0000A0	STCTQ	SF	C	R*4	000008				

***** COMMON INFORMATION *****

NAME OF COMMON BLOCK * THREE# SIZE OF BLOCK 00000C HEXADECIMAL BYTES

VAR. NAME	TYPE	REL. ADDR.									
KG	R*4	000000	J	R*4	000004	STCTQ	R*4	000008			

COMPILER GENERATED LABELS

LABEL	ISN	ADDR	LABEL	ISN	ADDR	LABEL	ISN	ADDR	LABEL	ISN	ADDR
100001	2	0000C8	100002	6	0000D4	100003	7	0000F6			

*OPTIONS IN EFFECT*NAME(LOADG) NOOPTIMIZE LINECOUNT(60) SIZE(MAX) AUTOUBL(NONE)

*OPTIONS IN EFFECT*SOURCE EBCDIC NOLIST NCDECK OBJECT MAP NOFORMAT GOSTMT XREF NOALC NOANSF NOTERM FLAG(I)

STATISTICS SOURCE STATEMENTS = 8, PROGRAM SIZE = 378, SUBPROGRAM NAME = LOADG

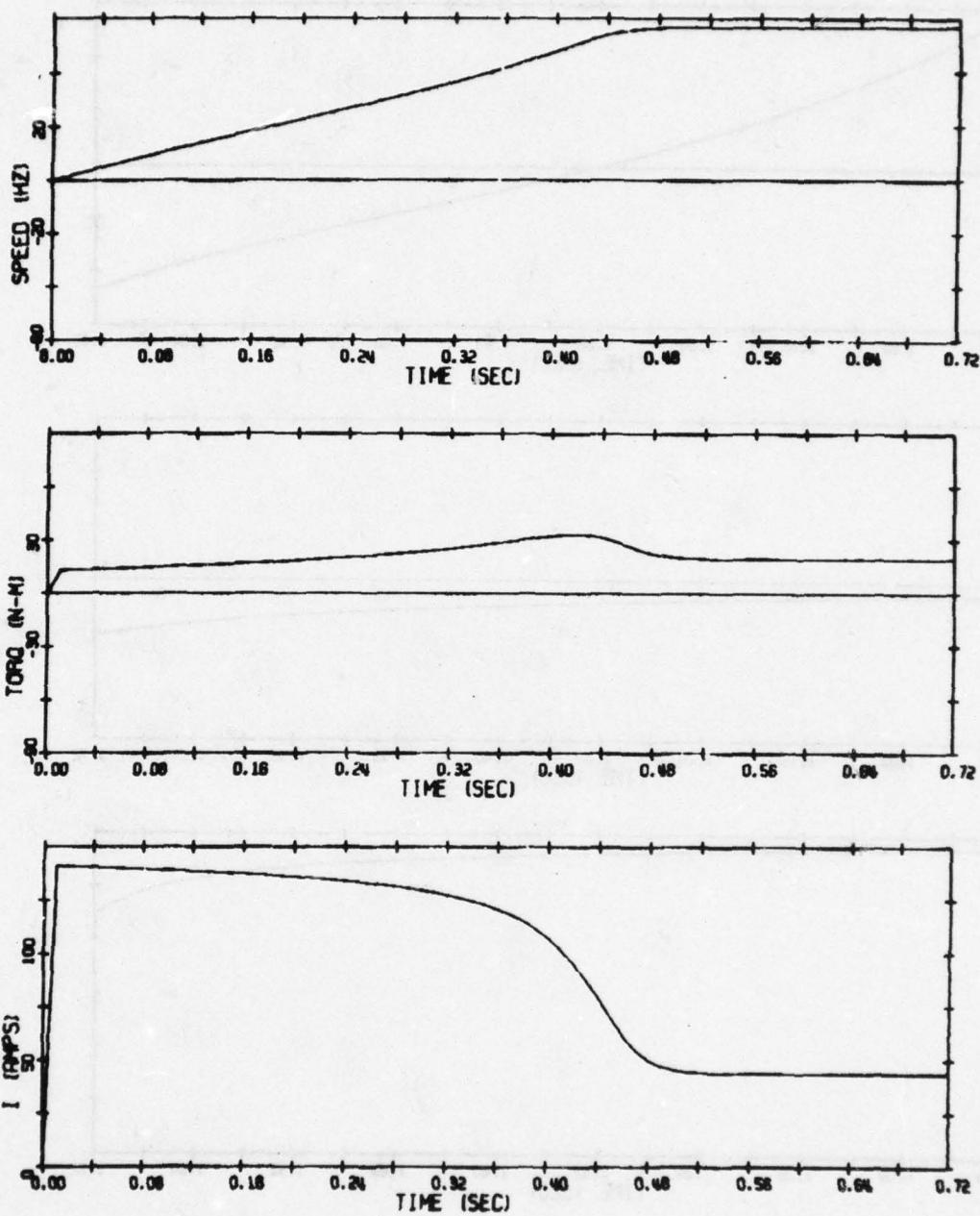
STATISTICS NO DIAGNOSTICS GENERATED

NO CONTROLLER

$f = 0\text{Hz}$ TO 60Hz STEP

STEP FUNCTION # 1

10/21/77

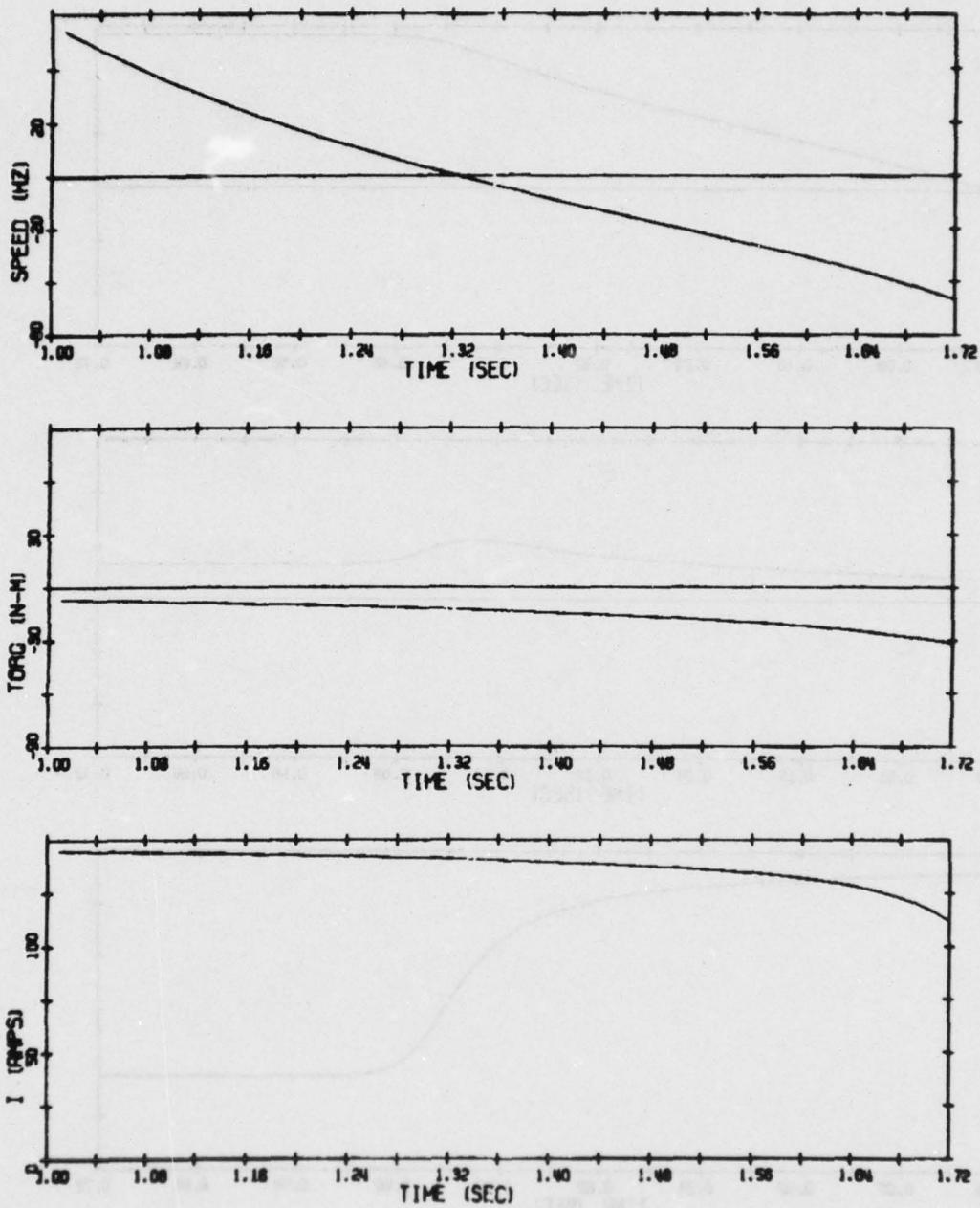


NO CONTROLLER

$f = > + 60\text{Hz}$ TO $- 60\text{Hz}$ STEP

STEP FUNCTION # 1

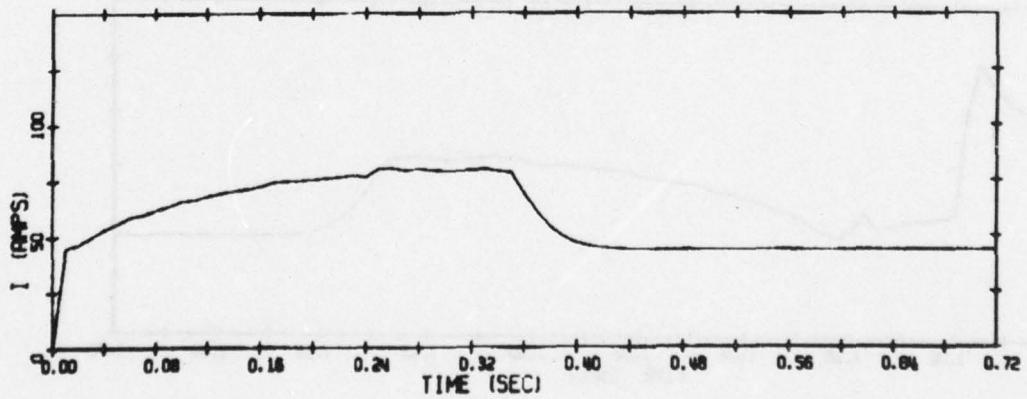
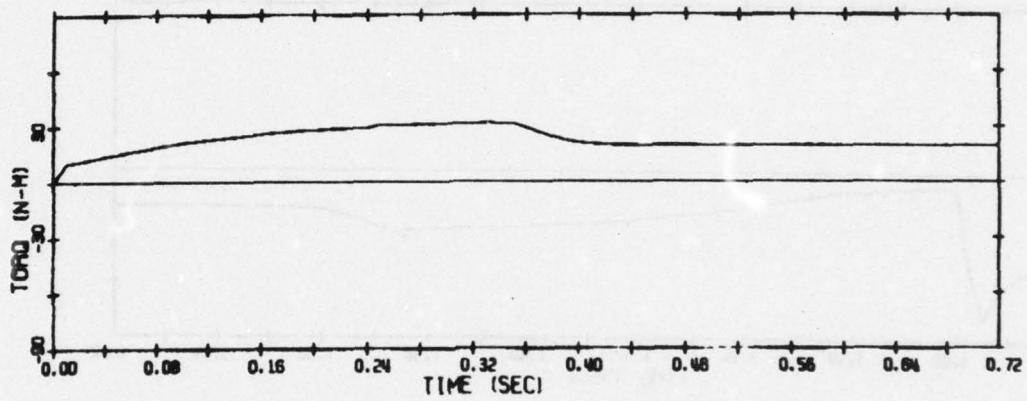
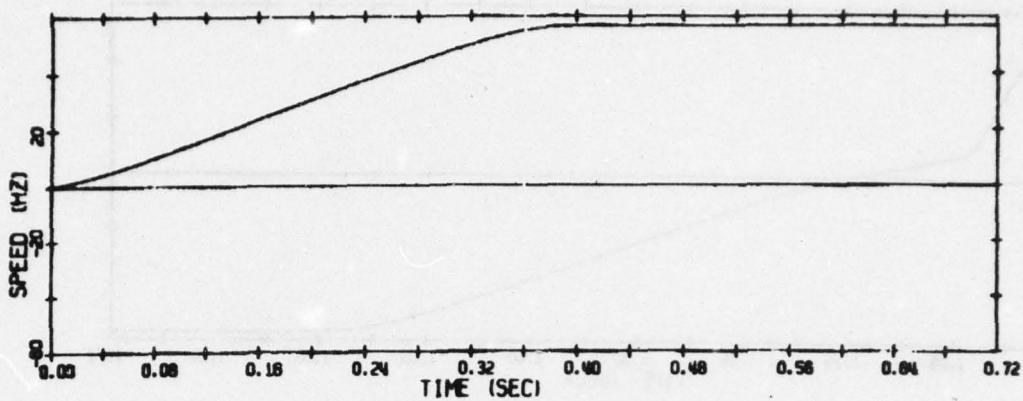
10/21/77



DLTAUP = 7.5HZ
DLTADW

STEP FUNCTION # 2

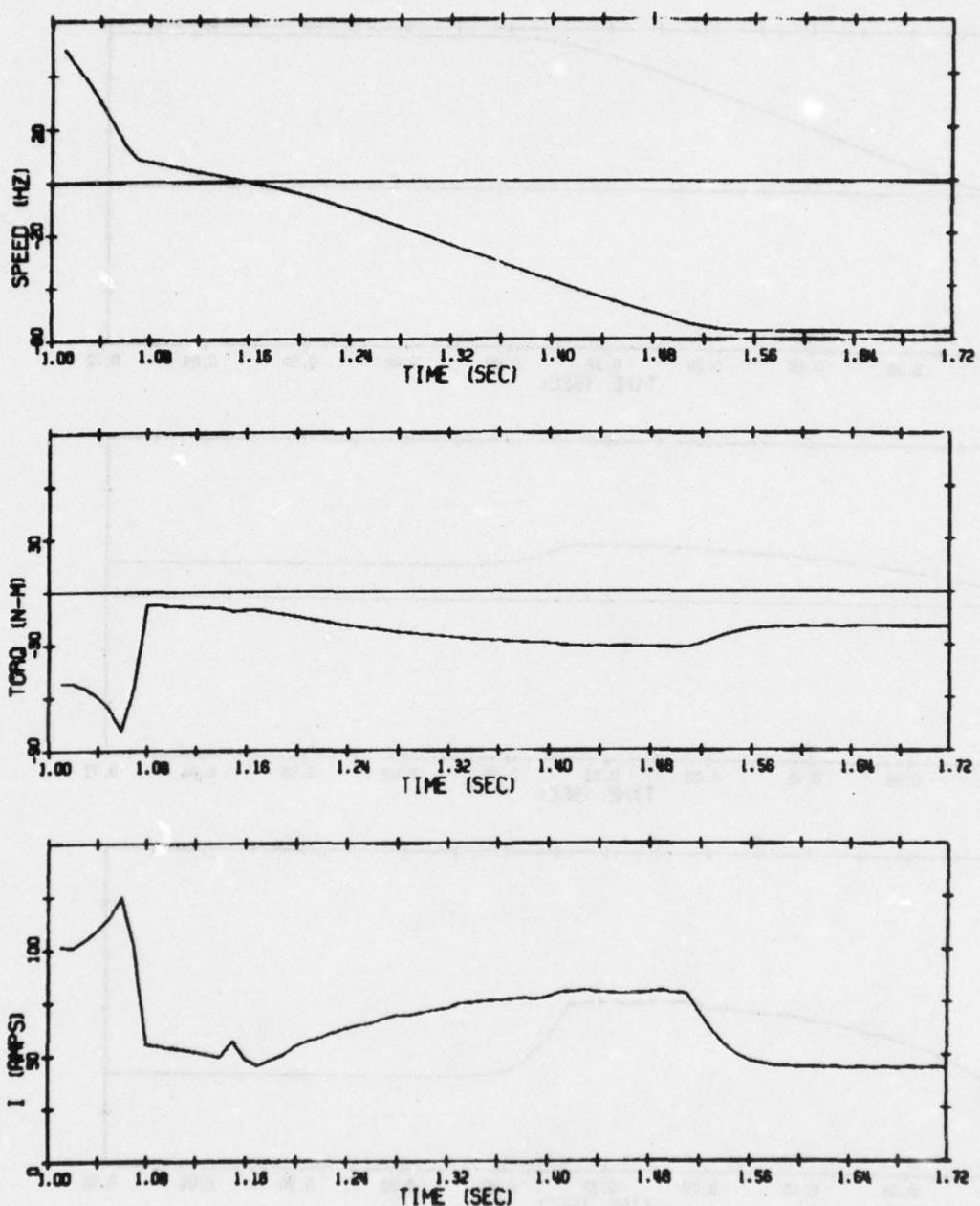
10/21/77



DLTAUP = 7.5HZ
DLTADW

STEP FUNCTION # 2

10/21/77



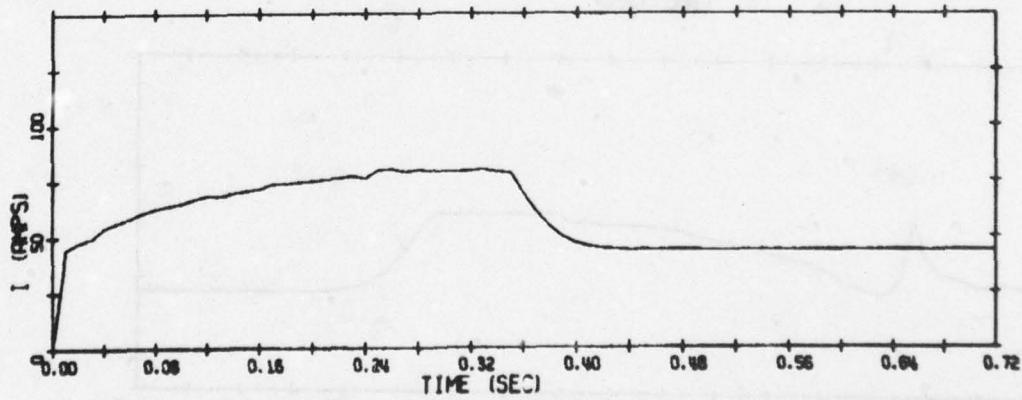
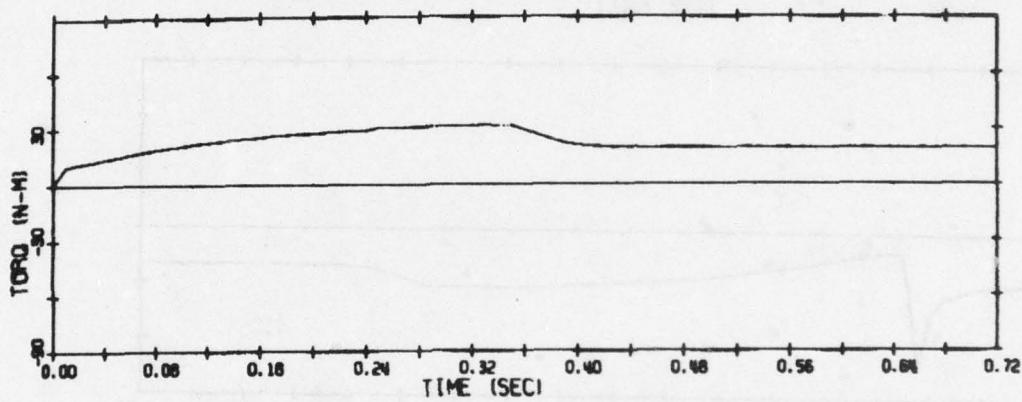
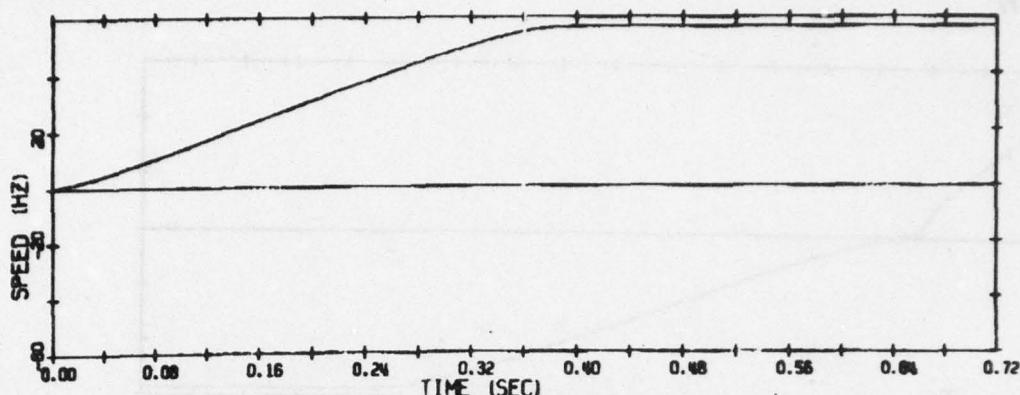
FINAL CONFIGURATION

DLTAUP = 7.5HZ

f START = TAC + 7.25HZ

STEP FUNCTION # 6

10/21/77

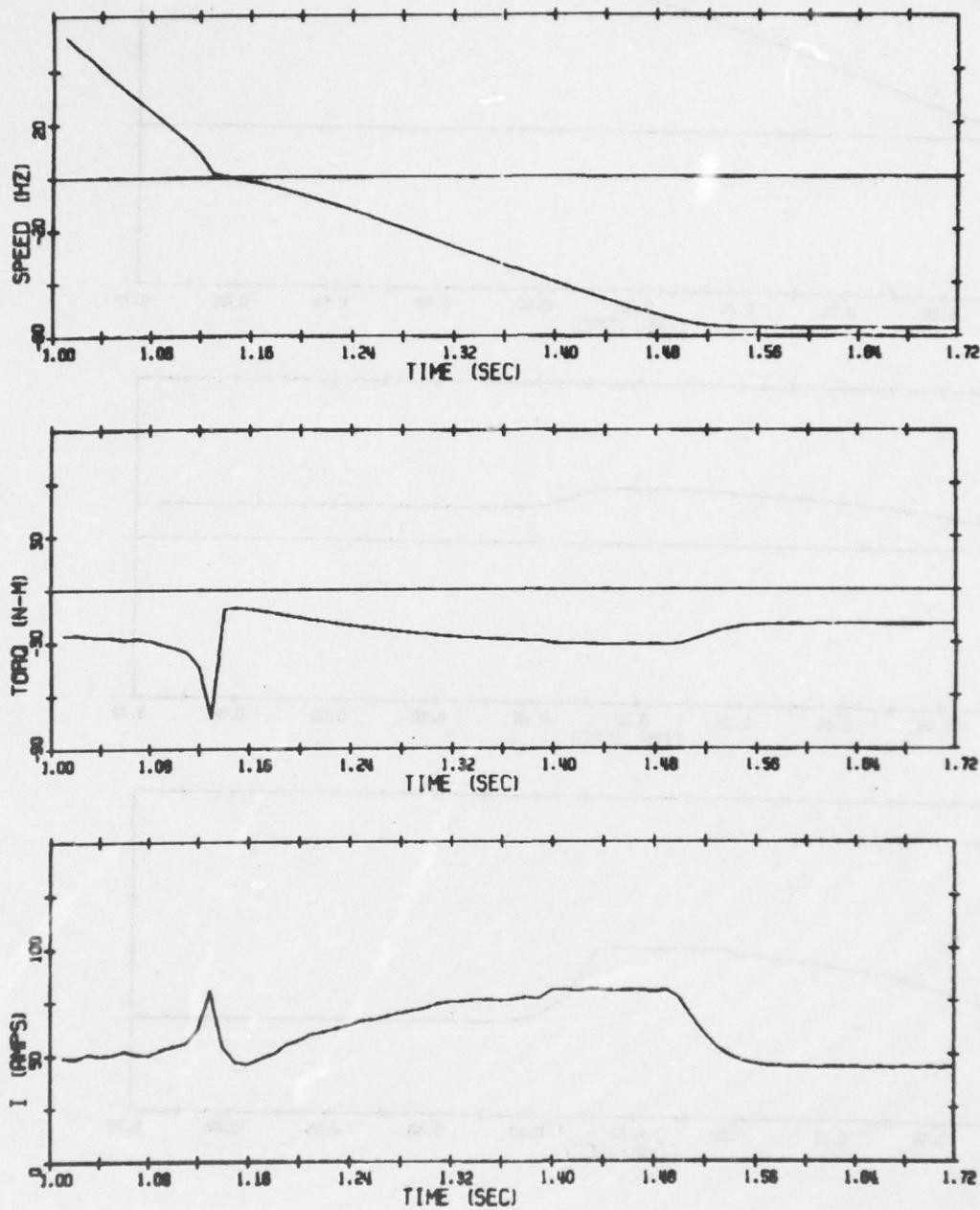


FINAL CONFIGURATION

DLTAUP = 7.5HZ
DLTADW = 3HZ
TLL 2HI = 5.5HZ
TLL 2LO = 5.0HZ
 $f_{REVERSE} = 7.25\text{HZ}$ (SIGNS DIFFERENT)
= TAC + 7.25HZ (SIGNS SAME)

STEP FUNCTION # 6

10/21/77



FOURIER COEFFICIENTS VRS % MODULATION

% MOD 1 PULSE MODE

10/24/77

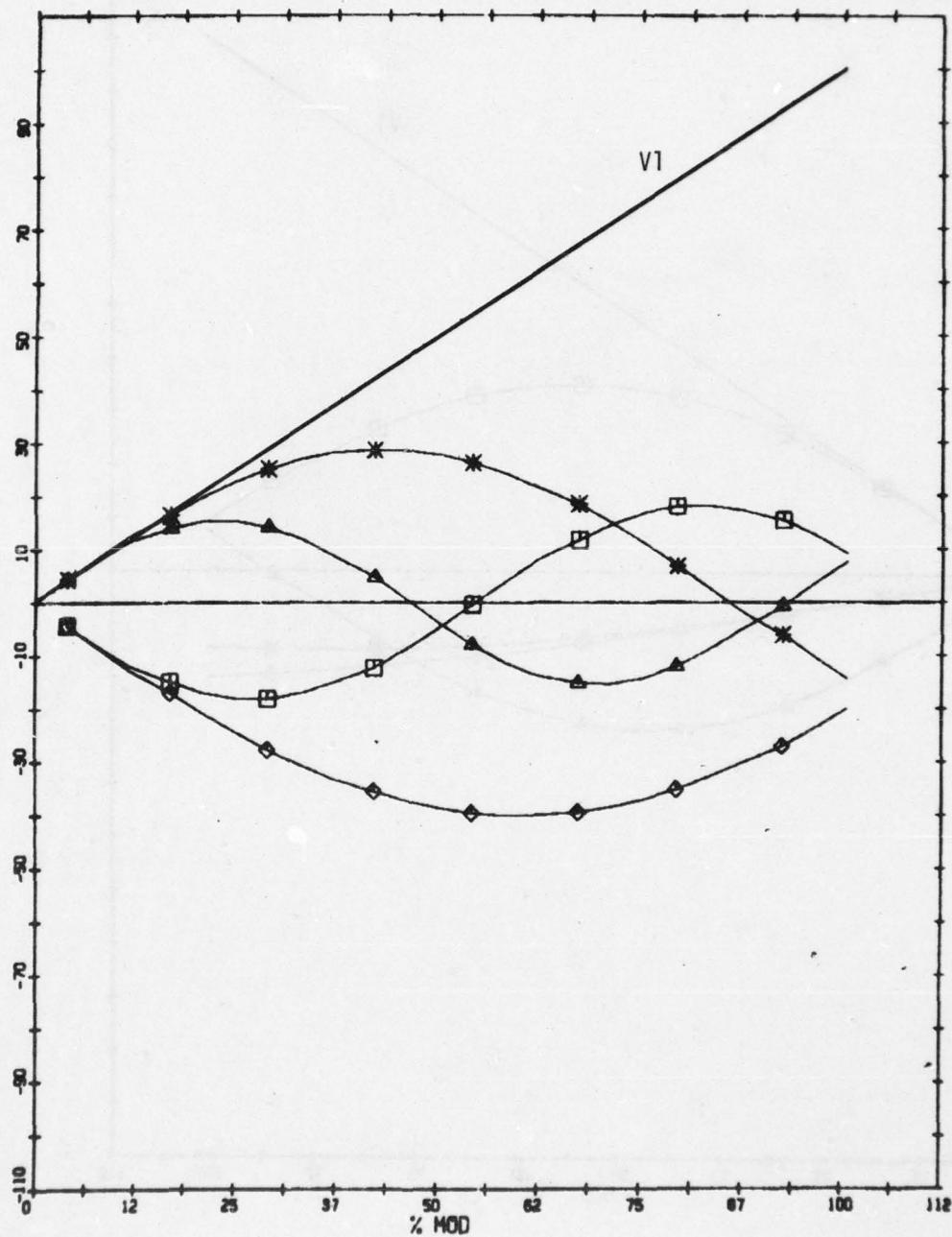
X % MOD

◊ V5

* V7

□ V11

▲ V13



% MOD 2 PULSE MODE

10/24/77

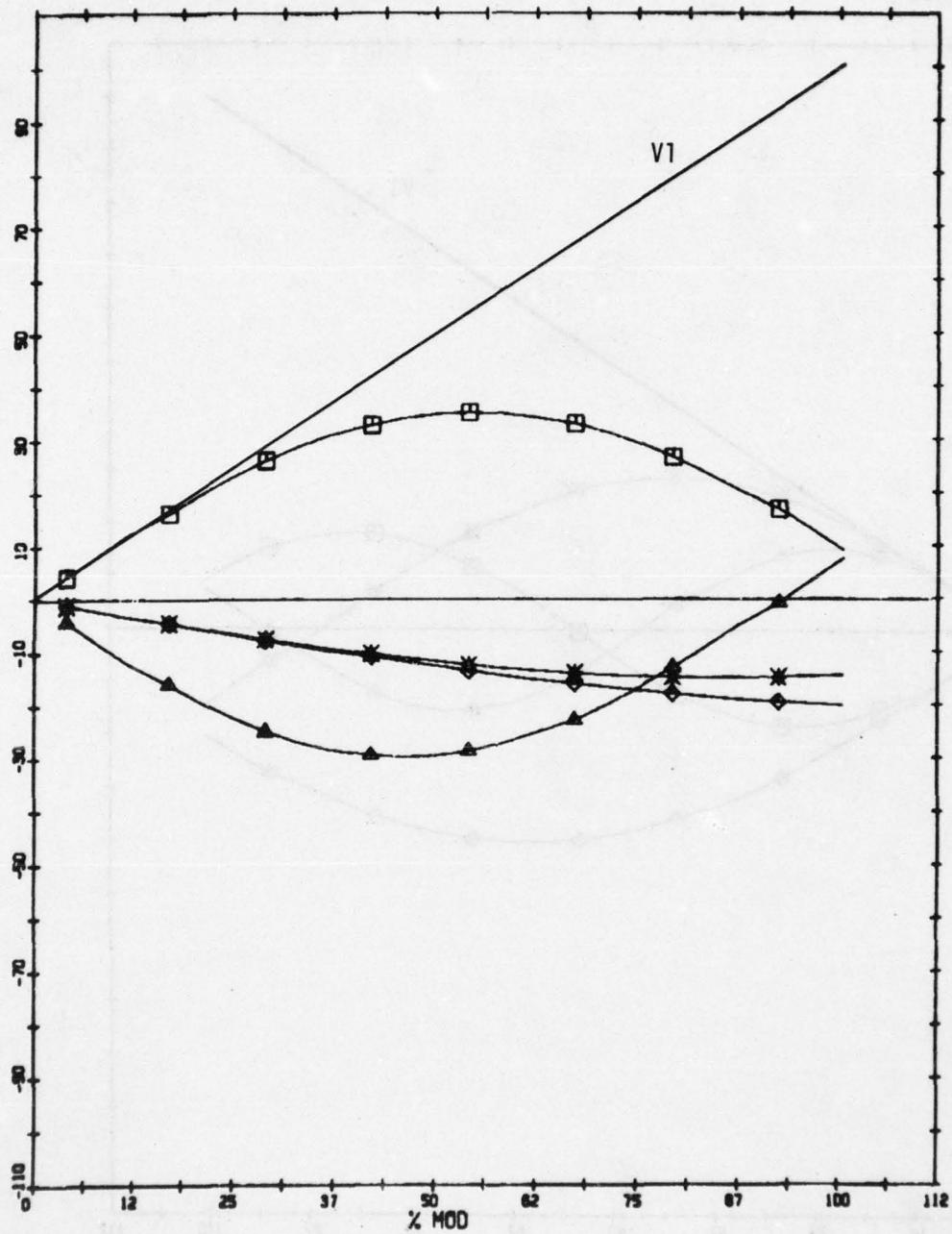
X % MOD

◊ V5

* V7

□ V11

△ V13



% MOD 4 PULSE MODE

10/24/77

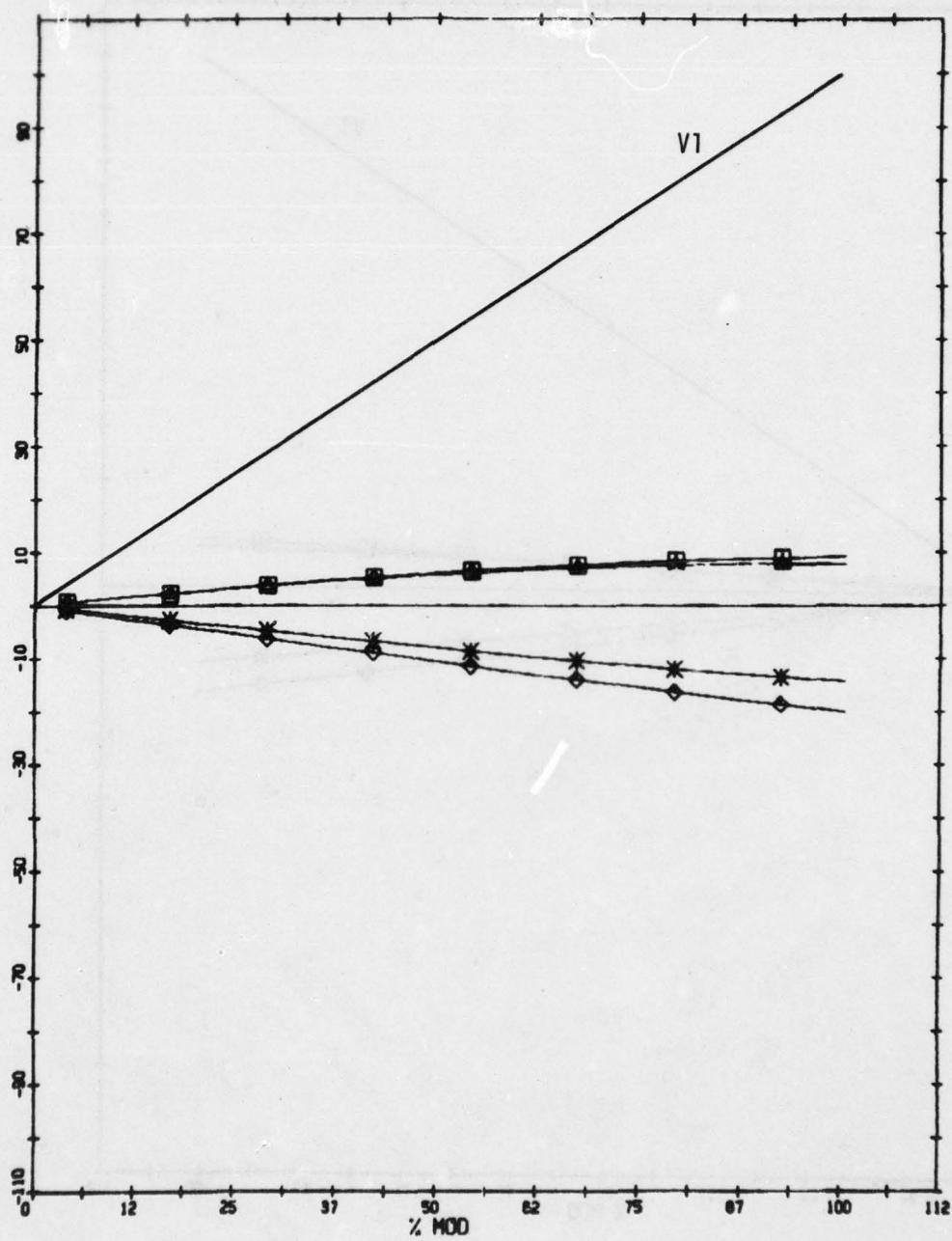
X % MOD

◊ V5

* V7

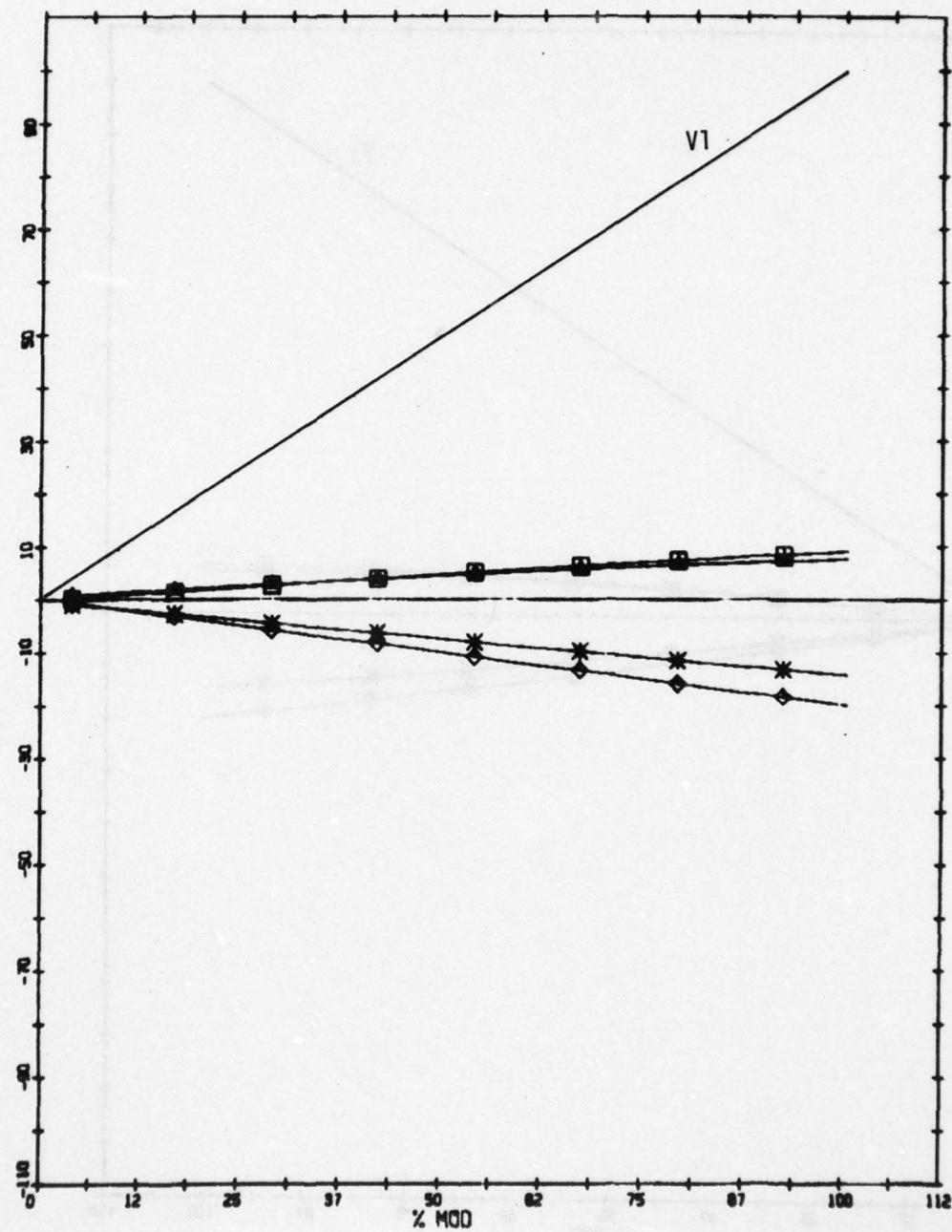
□ V11

△ V13



% MOD 8 PULSE MODE
10/20/77

X % MOD ♦ V5 * V7 □ V11 ▲ V13



% MOD 1 PULSE MODE

10/24/77

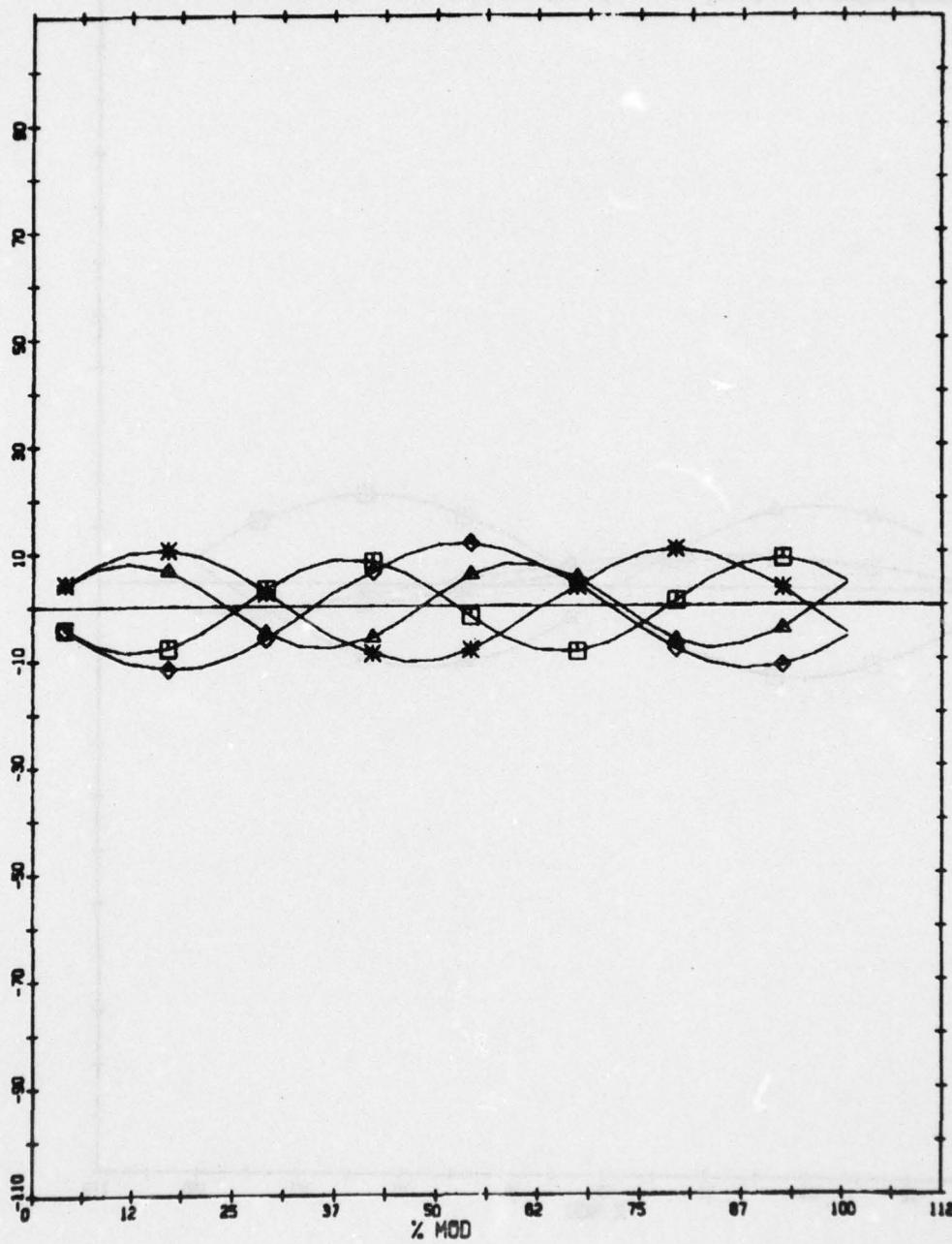
X % MOD

◊ V17

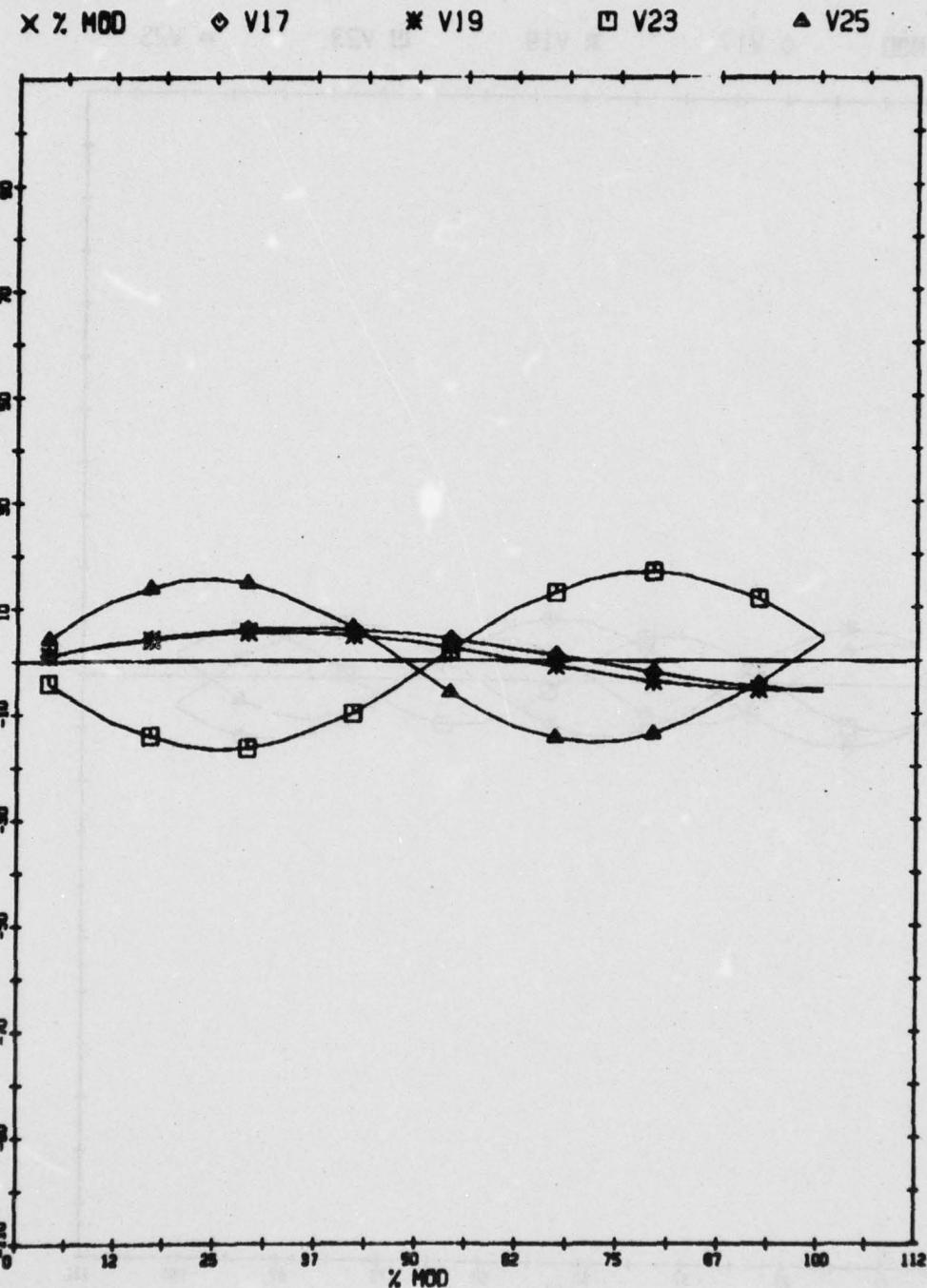
* V19

□ V23

▲ V25



% MOD 2 PULSE MODE
10/24/77



% MOD 4 PULSE MODE

10/24/77

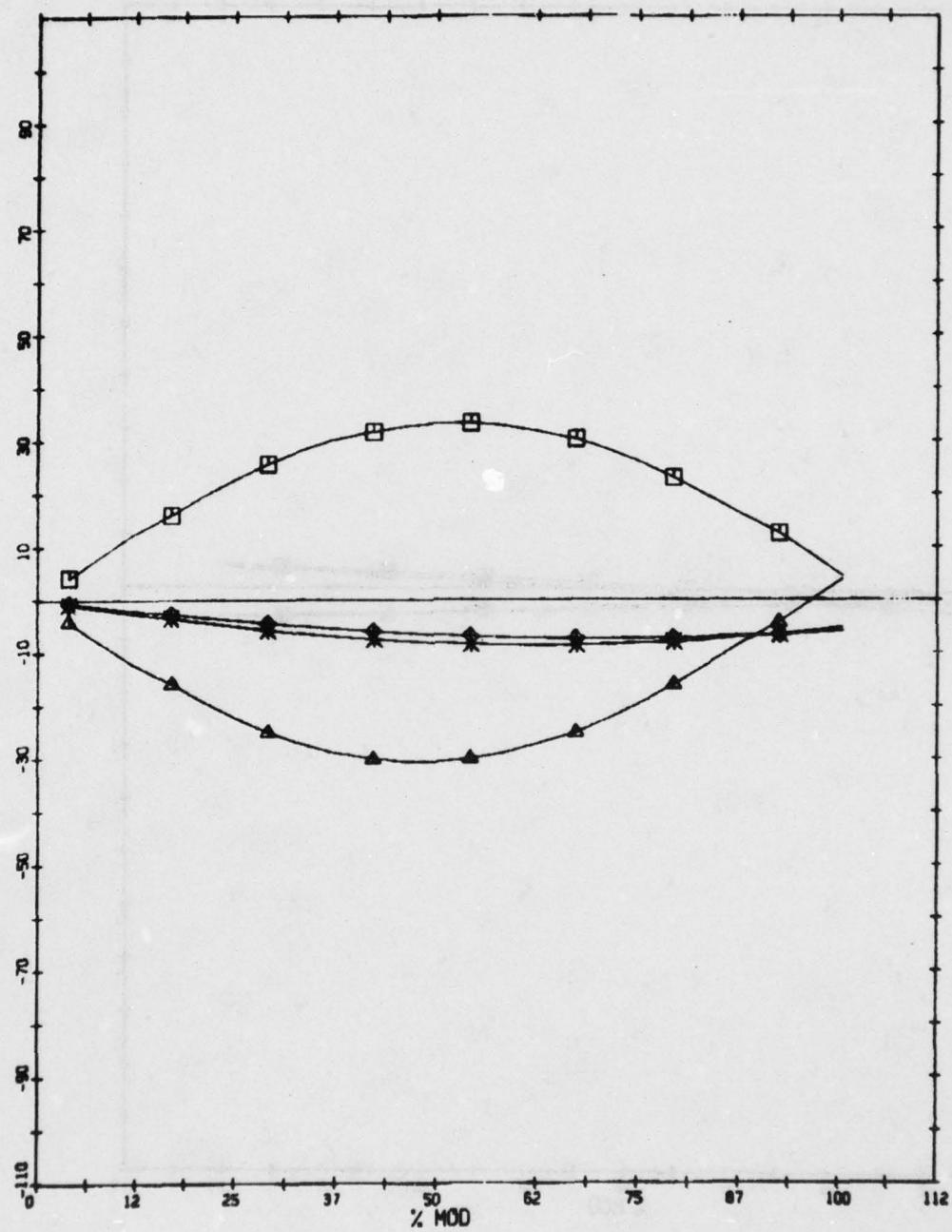
X % MOD

◊ V17

* V19

□ V23

△ V25



% MOD 8 PULSE MODE

10/24/77

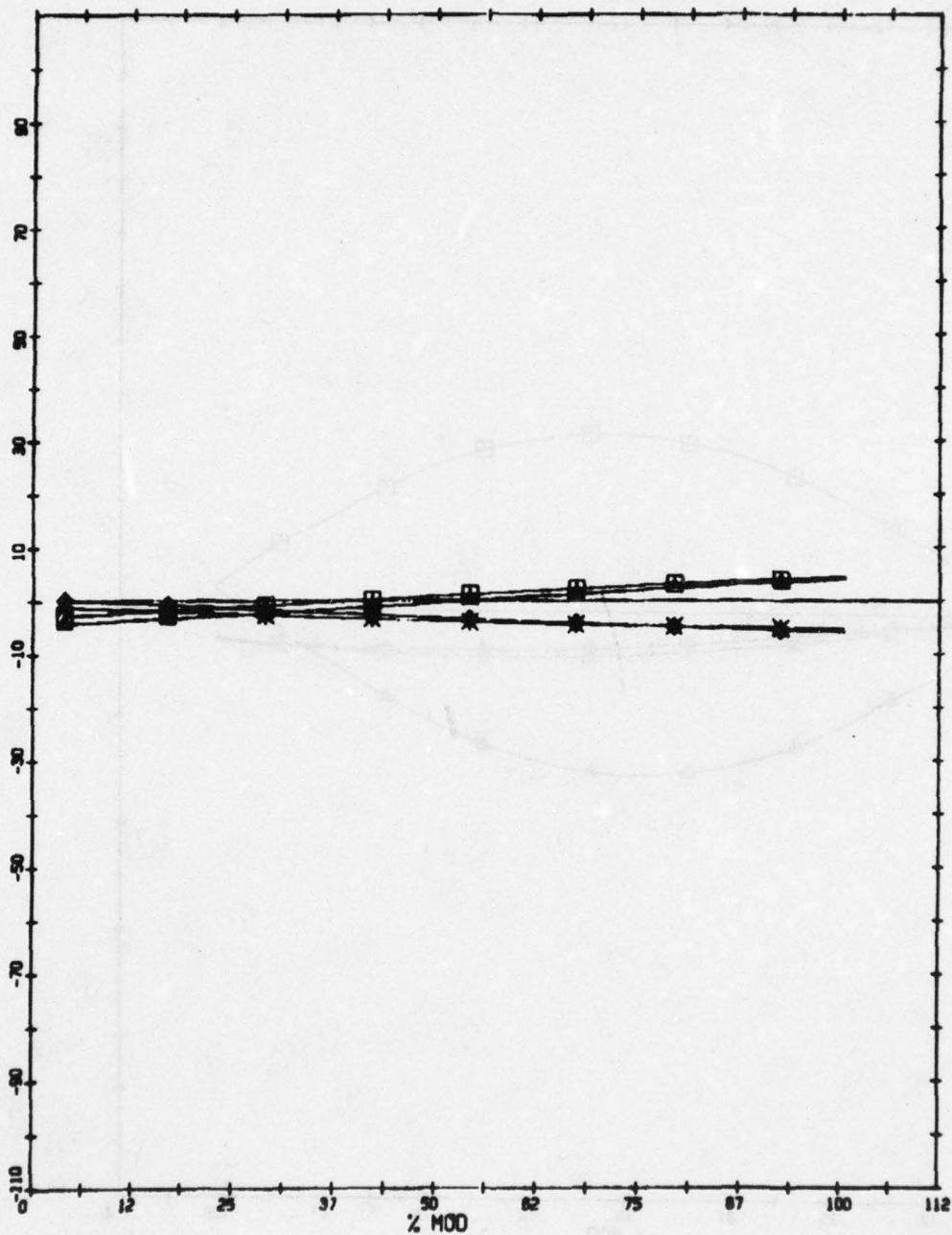
X % MOD

◊ V17

* V19

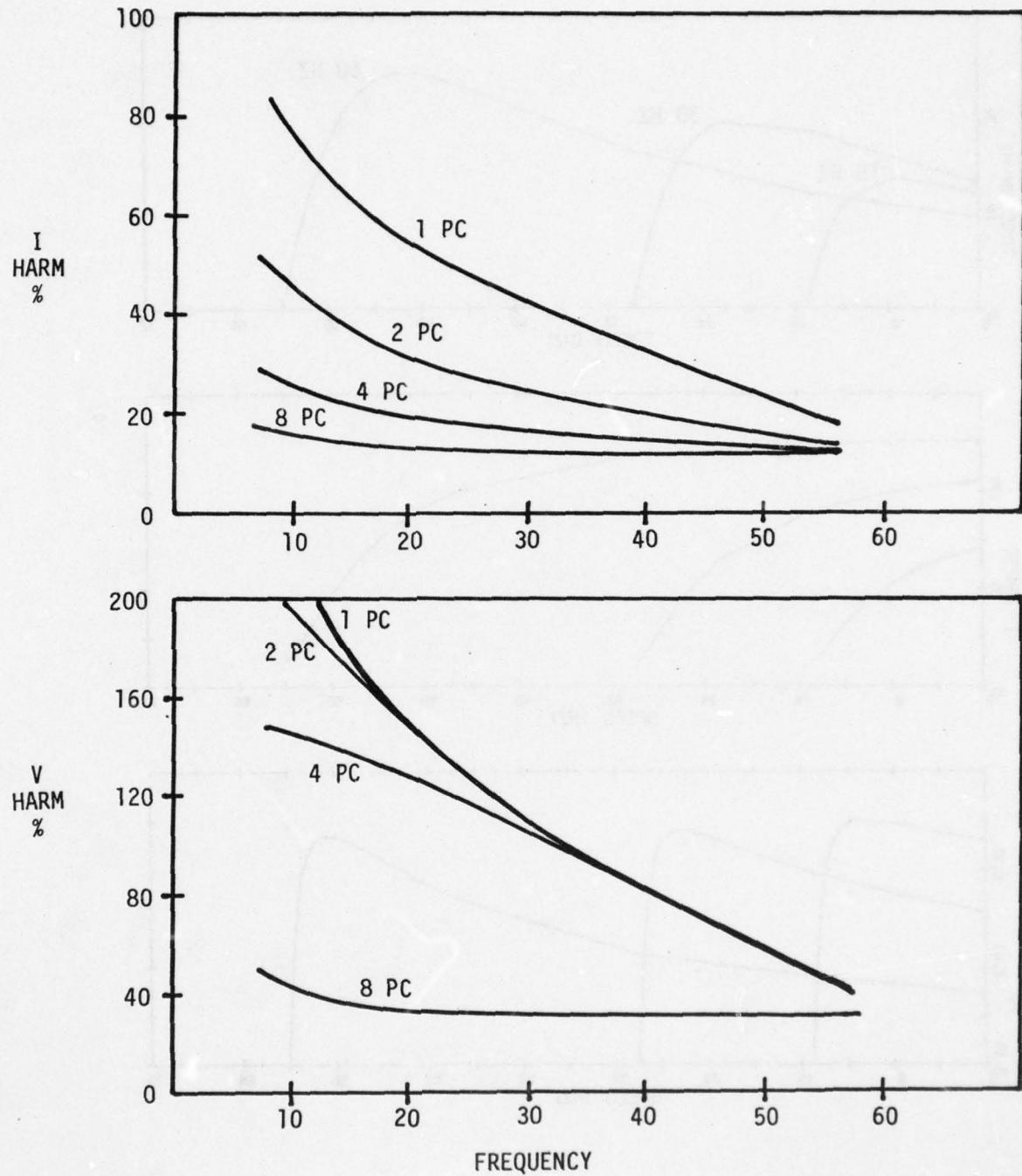
□ V23

▲ V25



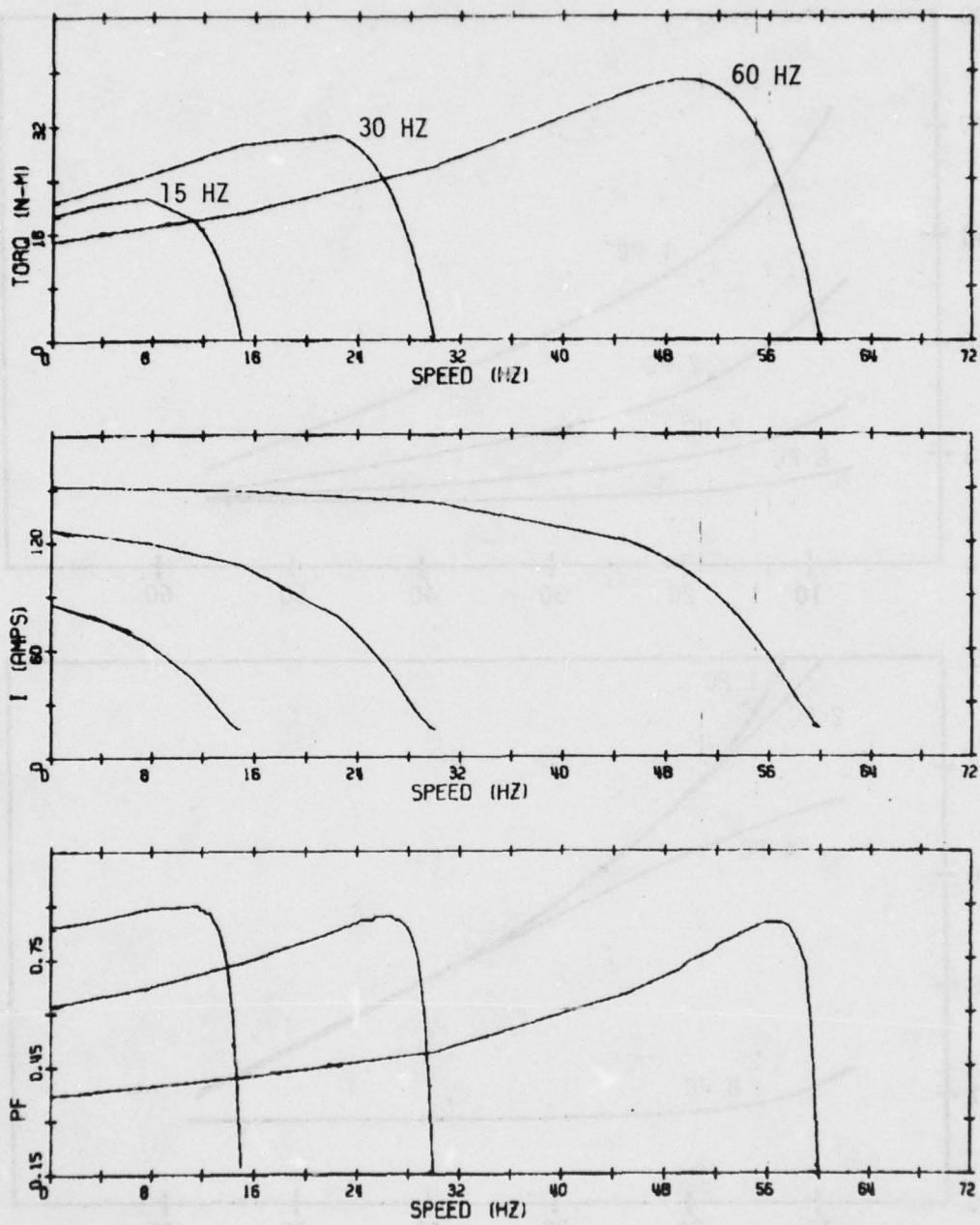
HARMONICS

10/21/77



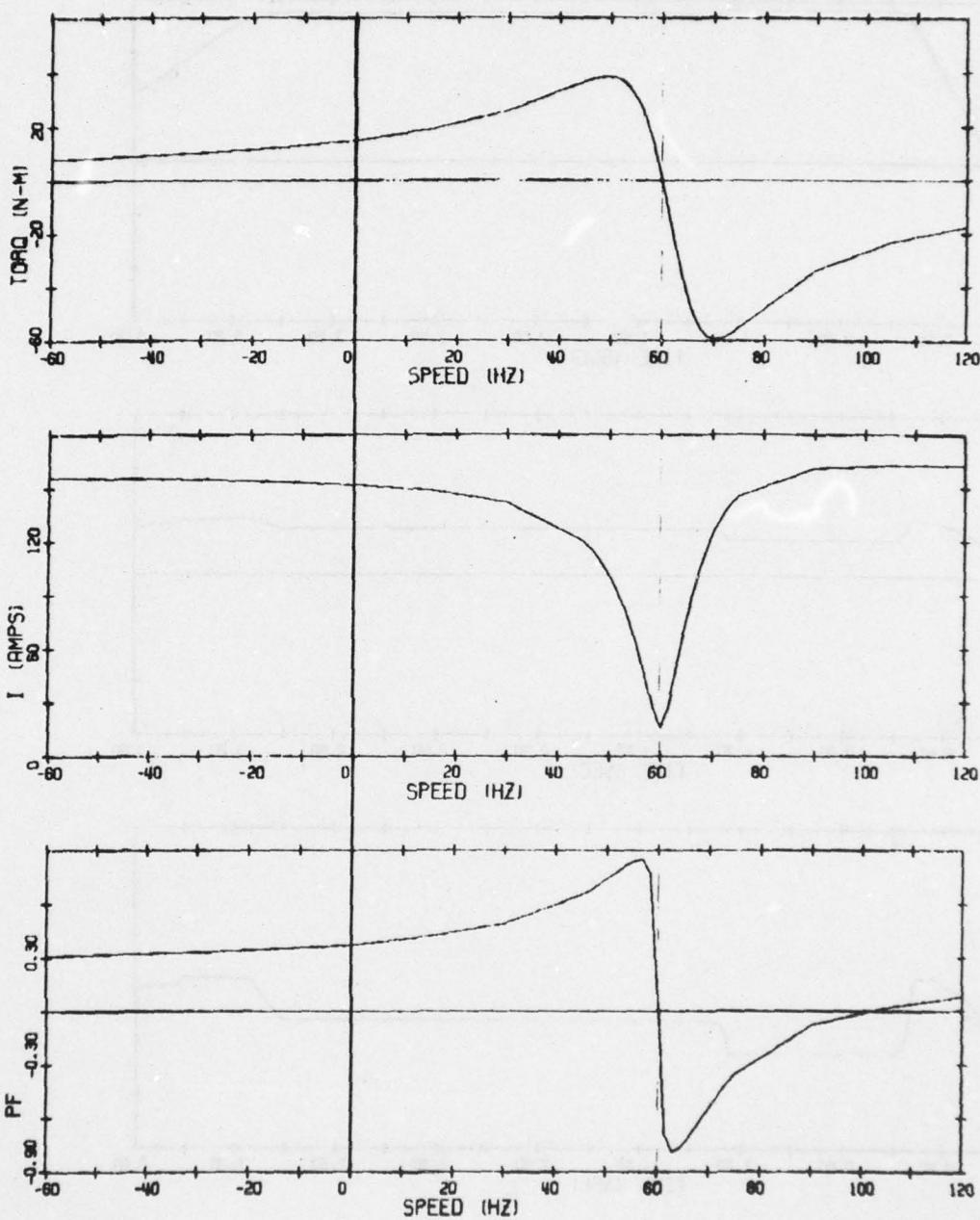
MOTORZ CURVES

10/17/77

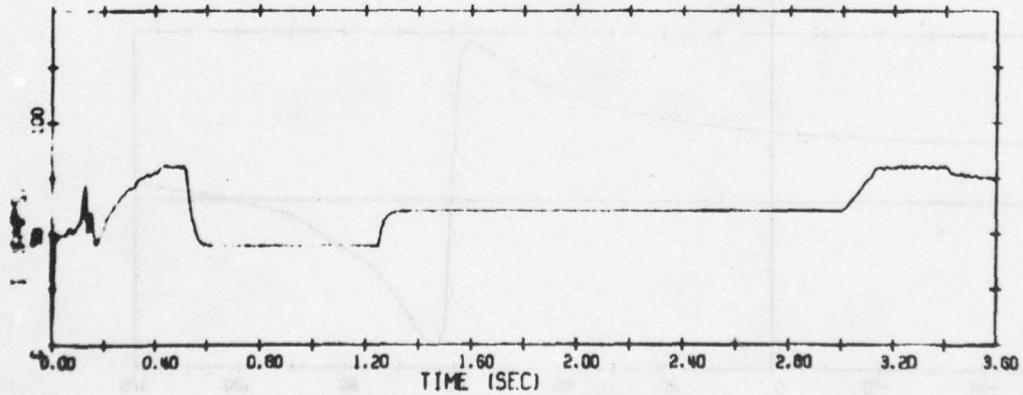
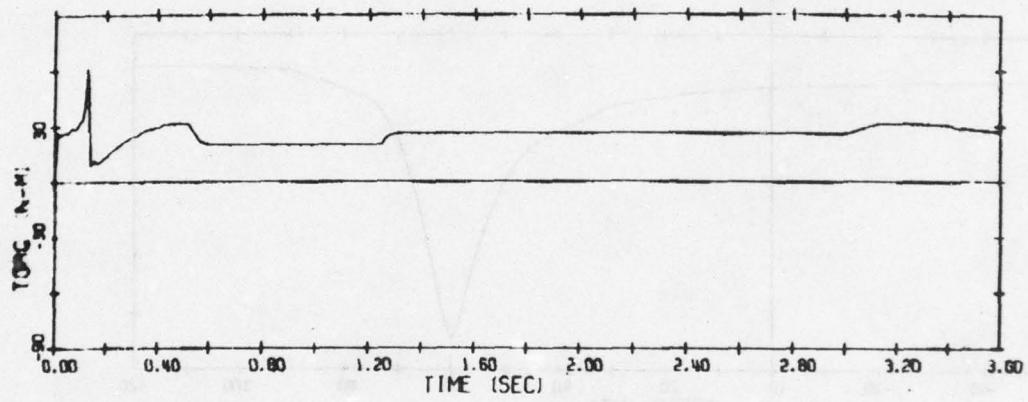
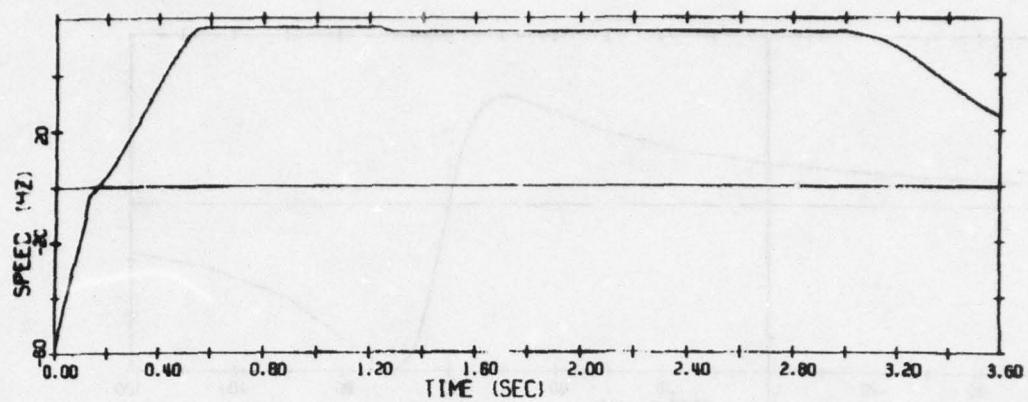


MOTORZ CURVES

10/17/77

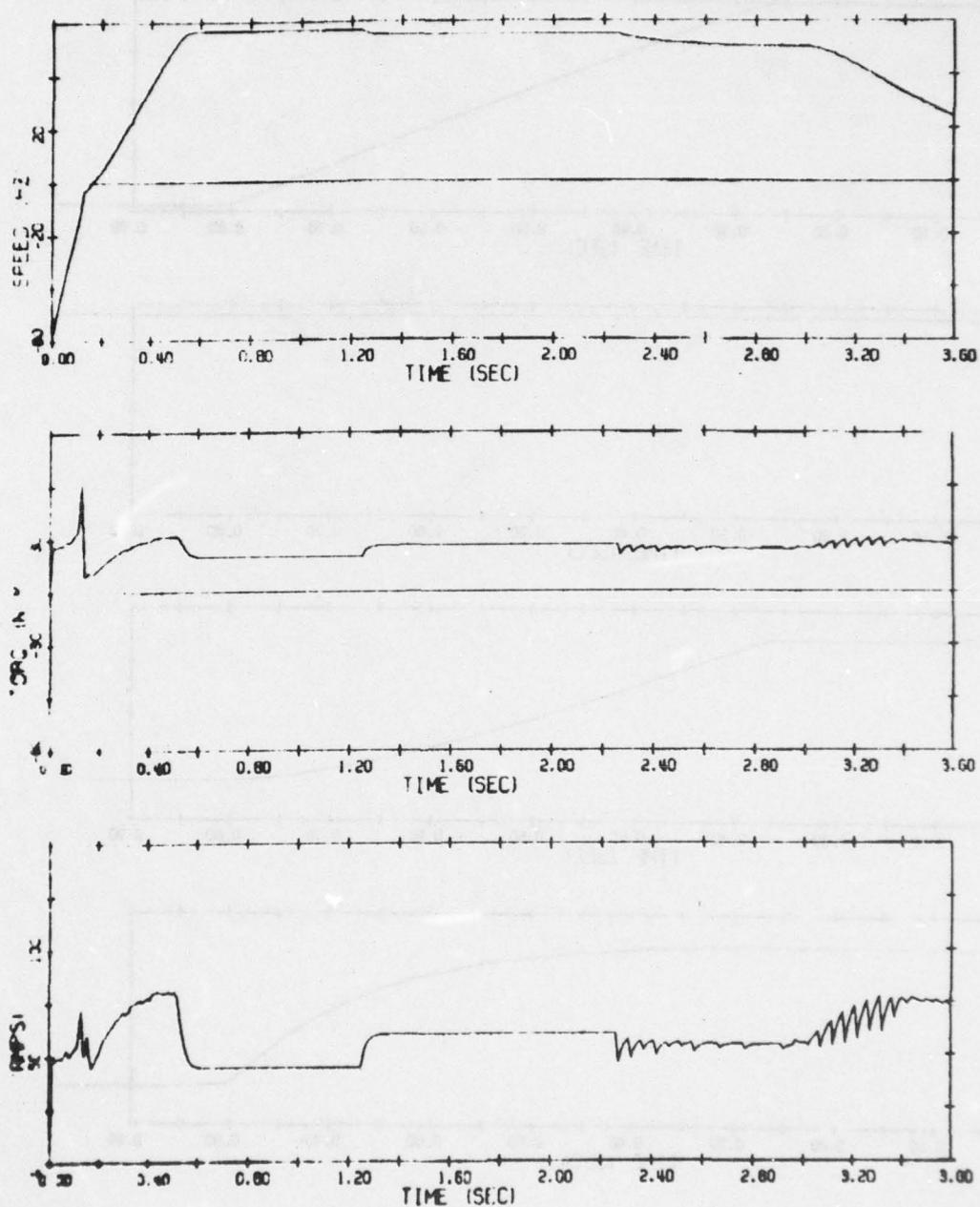


ILIMIT= .F.
11/7/77 INCREMENT SCFOLD



I LIMIT = 50 TO 100 AMPS

11/8/77 INCREMENT SCFOLD (50ms DELAY)



AD-A052 055

CHARLES STARK DRAPER LAB INC CAMBRIDGE MA
A MICROPROCESSOR CONTROLLED MOTOR CONTROLLER FEASIBILITY STUDY.(U)
MAR 78 A L GULOVSEN

F/G 9/5

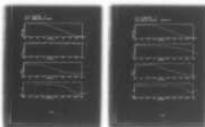
N00014-77-C-0352

NL

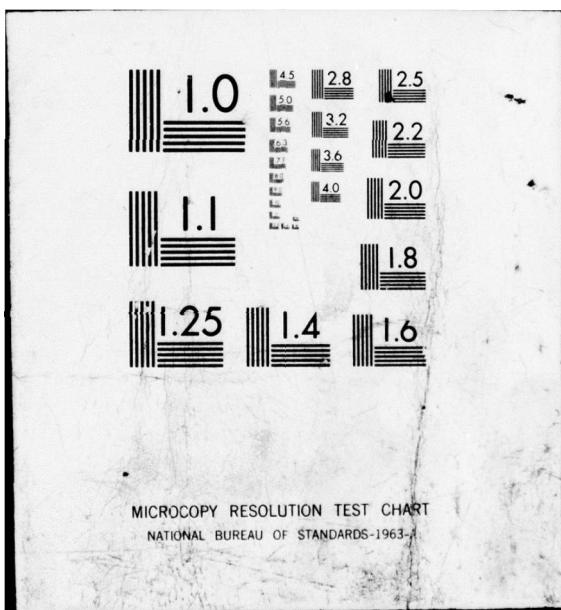
UNCLASSIFIED

R-1147

2 OF 2
AD
A052 055

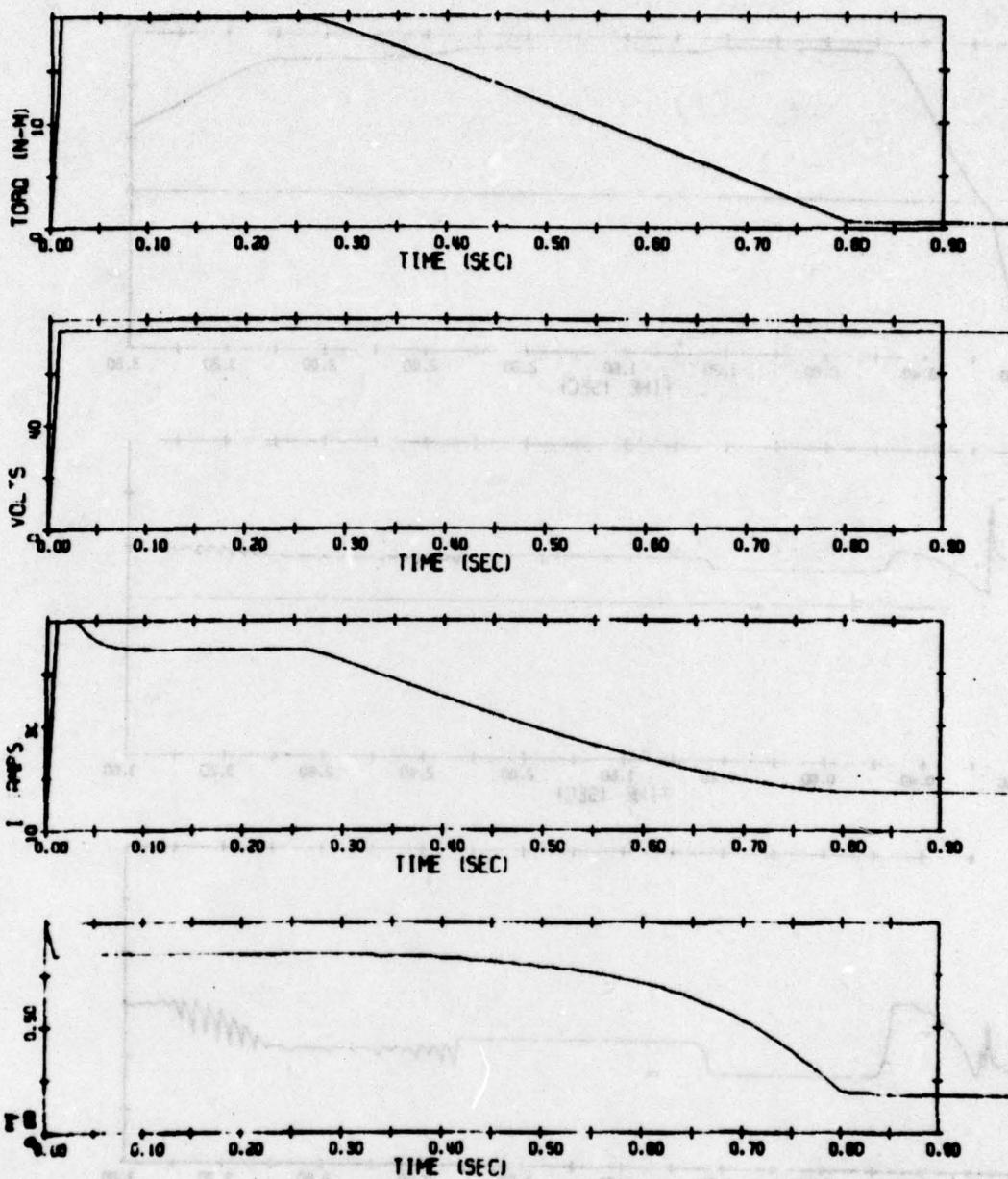


END
DATE
FILED
5-78
DDC



SLIP CONTROL .F.
11/8/77 INCREMENT/Delay SLIP CONTROL

295A 001 01 0E = TIMLI
C443C 001 01 0E = TIMLI
INCREMENT/Delay SLIP CONTROL



SLIP CONTROL .T.
11/8/77 SIDEAL=5HZ (INC4DELAY) (30ms DELAY)

